PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-305468

(43)Date of publication of application : 28.11.1997

(51)Int.Cl. G06F 12/00
G06F 12/00
G06F 9/44
G06F 13/00

(21)Application number : 08-120498

(22)Date of filing : 15.05.1996

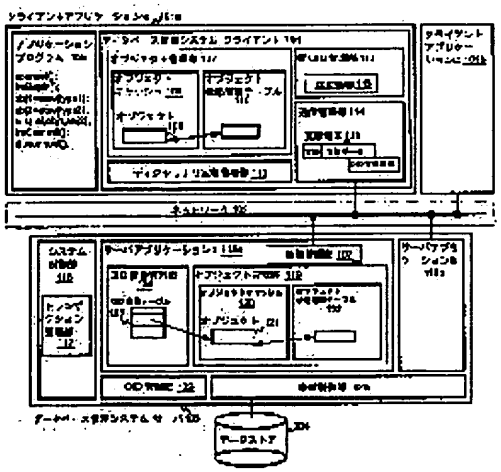
(71)Applicant : HITACHI LTD

(72)Inventor : KOBAYASHI KEN
YAMAMOTO YOICHI
ASAMI MASATO

(54) OBJECT MANAGING METHOD

(57)Abstract:

PROBLEM TO BE SOLVED: To provide an object managing method which can efficiently process object operation by a client application without communicating with a server while guaranteeing object discrimination in an object-oriented data base system constituted in a client-server type.



SOLUTION: An object management part 107 of a database management(DBMS) client 106 allocates a temporary object identifier(OID), based upon a serial number (generation serial number 113) of object generation order, to a generated object 108. To an update message 115 that the database management (DBMS) client 106 generates, OID conversion information showing the position of the temporary OID in the message is added. A database management(DMMS) server 103 stores the generated object so that access efficiency of a data store 104 is improved, and converts the temporary OID into a format OID which is unique in the system according to the OID conversion information.

LEGAL STATUS

[Date of request for examination]	05.03.2001
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] This invention relates to the object management approach, and relates to the suitable object management approach to use for the object-oriented database system which consists of client-server molds especially.
[0002]
[Description of the Prior Art] The system configuration of the client-server mold through a network is applied to many computing systems as a means for having availability and maintainability. Also in an object oriented database (Object-Oriented Database:OODB) system, it is common to apply the system configuration of a client-server mold. In this case, a database management system (Database Management System:DBMS) is made into a server, and the application program (Application Program:AP) which accesses the data managed with a database management system (DBMS) is made into the client.
[0003] Generally there are a "dynamic condition" and "a permanent condition" in the object treated by the object oriented database (OODB) system. The "dynamic condition" of an object has usually pointed out the condition of having set on the primary storage of a computer so that an application program (AP) can operate it to an object. The "dynamic condition" has the volatilization-property. When the contents of the primary storage are lost, a "dynamic condition" is lost by halt of application program (AP) process killing and database system etc.
[0004] "A permanent condition" is in the condition evacuated in order to reproduce a "dynamic condition", and it is usually held at secondary storages, such as a disk. In addition, it is called "activation" (activate) to move object data from "a permanent condition" to a "dynamic condition." Moreover, the condition which can be operated from an application program (AP) is called "activated state" in the "dynamic condition."
[0005] The object perpetuity service specification (Object Persistence Service Specification (OPSS)) specified by the object management group company (Object Management Group Inc. (OMG)) is known as standard specifications for making "the permanent condition" of an object possible, namely, realizing perpetuity of an object. An object perpetuity service specification (OPSS) is a guideline which gives the device and interface for managing the permanent condition of an object. For example, according to the object perpetuity service specification (OOPS), a datastore (data store) means the device in which the storage for the permanent condition of an object is given, and a permanent identifier (Persistent Identifier:PID) means the value holding the information which identifies the memory location of the permanent condition of an object.
[0006] When dealing with two or more objects dynamically and permanently, the device in which each object is uniquely identified within a system is needed. According to the data model of the standard specifications of the object oriented database (OODB) which an object database management group company (Object Database Management Group (ODMG)) specifies "The ObjectDatabase Standard:ODMG-93 Release 1.1", Cattel, R.G.G.ed.Morgan As indicated by KaufmannPublishers and

1994 In order to identify each object in database system uniquely, an object identifier (Object Identifier: OID) is used.
[0007] The "relation" (relationship) between objects is the means of the features-modeling in an object-oriented data model. According to a general object design technique called "OMT", "relation" is a means for establishing the relation of an object, and a "link" (link) is physical or notional association between objects as indicated by Eiichi Hanyuda supervision of translation besides "object-oriented methodology OMT modeling, a design", and J.Rumbaugh, and TOPPAN (1992). "Relation" is the assembly (definition) of a "link" with the same semantics.
[0008] As the mounting approach of a general related (link) function, there is a method of holding an object identifier (OID) inside object data. The object corresponding to a meaning is interpreted as the associated link place object to the object identifier (OID) currently held inside data.
[0009] Relation essentially has bidirection. For example, when defining "it relating " working between firm"" with "people", the reverse relation "it employs" of "working" is the relation between a "firm" and a "man", following which direction also has semantics equally and the same relation in the bottom is shown.
[0010] Moreover, there is a concept of "multiplicities", such as many to many, in relation, and it is shown how many objects are associated by the link. For example, when a "firm" "employs" two or more "employees", the multiplicity of this relation serves as one-pair **.
[0011] In the conventional object oriented database (OODB) system, it guarantees that an object identifier (OID) serves as a meaning within a system. Therefore, issue management of an object identifier (OID) is performed by the database-management-system (DBMS) server which manages the whole object oriented database (OODB) system.
[0012] Therefore, in order to assign an object identifier (OID) to the object generated with client application, the communication link with a database-management-system (DBMS) server is needed.
[0013] On the other hand, when managing the link between objects, generally an object identifier (OID) is used as information on a link place. Therefore, in case a link is set as a new object, after telling object generation previously to a database-management-system (DBMS) server and assigning an object identifier (OID), link setting processing must be performed.
[0014]
[Problem(s) to be Solved by the Invention] As a method which performs this processing of a series of briefly, object generation is first communicated with a database-management-system (DBMS) server, the object identifier (OID) which published and published the object identifier (OID) is received, a link setup is performed by the client, and it is possible about the updating demand accompanying a link setup to communicate with a database-management-system (DBMS) server separately.
[0015] However, it does in this way and there is a problem that the count of a communication link increases, by the method which manages a link. Since processing cost is high, in the above mode of processing, in a client-server system, a communication link cannot process the demand of a client efficiently, is highly efficient and cannot operate.
[0016] As an example of the client-server mold object oriented database (OODB) system which treats the link between objects efficiently, the object store (Object Store) is known as a product of for example, an object design (Object Design) company. At the object store, the pointer on memory is used about discernment of an object, and the link place is directly expressed with the pointer about the relation between objects. The approach called the "virtual-memory mapping architecture" adapting paging of an operating system (OS) and the device of virtual memory is adopted so that this pointer can be used permanently.
[0017] By this approach, the storing configuration of the object in a data page is locally determined by the client, and the condition of that page is stored in a datastore as it is in a server. Therefore, improvement in the access efficiency in processing of each client can be aimed at by storing in a near location physically the object treated for every client collectively in a datastore (clustering).
[0018] When performing object actuation regardless of processing of each client with such a storing configuration, however, for example When all affair retrieval of the object which has a certain DS for

the whole database (it belongs to a certain type) is carried out, There is much access to the data page of the physically distant location, especially when special devices, such as an index for retrieval, are not used, all data pages must be made applicable to retrieval, and effectiveness is bad.
[0019] It is difficult for two or more clients to carry out storing control of the object group treated according to an individual collectively by the approach by "virtual-memory mapping architecture."
[0020] The purpose of this invention is in the object-oriented database system which consists of client-server molds to offer the object management approach that object actuation with client application can be processed efficiently, without communicating with a server, guaranteeing object epicritic within a system.
[0021]
[Means for Solving the Problem] Since the above-mentioned purpose is attained, this invention a database-management-system client The object generated with client application is managed by temporary temporary object identifier. The database-management-system server carried out a reflection demand about the above-mentioned object Change the above-mentioned temporary object identifier into the formal object identifier which serves as a meaning with database system, control storing in the datastore of the above-mentioned object, and by the approach of starting Object actuation with client application can be processed without performing the communication link with a server.
[0022] In the above-mentioned object management approach, preferably, in case the above-mentioned database-management-system client carries out a reflection demand about the above-mentioned object to the above-mentioned database-management-system server, the additional object identifier conversion information for changing into the information which directs a reflection demand from the above-mentioned temporary object identifier at the above-mentioned formal object identifier is made to add, nd it can carry out conversion to a formal object identifier efficiently by the approach of starting.
[0023] In the above-mentioned object management approach, preferably, the above-mentioned database-management-system client manages the above-mentioned object by the number of the order of generation, it assigns the above-mentioned temporary object identifier to each object based on the number of this order of generation, and it becomes easy by the approach of starting to manage [of a temporary object identifier] it.
[0024] In the above-mentioned object management approach, preferably, the above-mentioned database-management-system client transmits two or more objects to the above-mentioned database-management-system server collectively, is made to carry out a reflection demand, and can reduce the count of a communication link between client-server by the approach of starting.
[0025] In the above-mentioned object management approach, a link is set up among two or more desirable above-mentioned objects, and the partial update information which sets link place information as the information which directs the above-mentioned reflection demand is added.
[0026]
[Embodiment of the Invention] Hereafter, the object management approach by 1 operation gestalt of this invention is explained using drawing 1 thru/or drawing 19 . Drawing 1 is the object oriented database (OODB) structure-of-a-system Fig. which applied the object management approach by 1 operation gestalt of this invention.
[0027] Through a network 102, the client applications 101a and 101b communicate with the database-management-system (DBMS) server 103, and constitute the system from a client-server mold which accesses the object stored in a datastore 104.
[0028] Without increasing the count of a communication link, this system is constituted so that object actuation (they are object generation and an object link setup especially) by the client can be performed. And it has the following (1) - (3) as fundamental mode of processing for avoiding a communication link.

[0029] (1) The issue client applications 101a and 101b of a temporary object identifier (temporary OID) publish and assign a temporary object identifier (OID) to the newly generated object, without communicating with the database-management-system (DBMS) server 103. here -- this temporary bject identifier (OID) -- "-- it will be called temporary OID."

[0030] this -- "-- temporary OID" is an object identifier (OID) published to a forward type so that the database-management-system (DBMS) server 103 may become a meaning within a system -- "-- it uses as contrasted with formal OID."
[0031] temporary -- the client applications 101a and 101b incorporate the serial number of the order of generation to the in-house data of an object identifier (OID), and OID creates it.
[0032] in the client applications 101a and 101b, the contents of the value and actuation demand which identify the object used as the candidate for object actuation are temporary -- it is recorded using OID.
[0033] (2) Transmit a reflection demand to the database-management-system (DBMS) server 103 corresponding to the object actuation in package transmission of updating reflection information, and the addition client applications 101a and 101b of OID conversion information collectively ignited by a transaction commitment.
[0034] temporary to the communication link wording of a telegram of a reflection demand -- formal in OID -- the additional information (OID conversion information) for changing into OID efficiently is added.
[0035] (3) an object identifier (OID) is changed and the reflection database-management-system (DBMS) server 103 is temporary based on the generation serial number of an object according to the above-mentioned OID conversion information to a datastore -- efficiently formal in OID -- change into OID and reflect the object actuation demand in temporary OID in communication link wording of a telegram in a datastore.
[0036] The database-management-system (DBMS) server 103 controls storing in a datastore, and it aims at improvement in access efficiency by clustering etc.
[0037] Next, the configuration of the client applications 101a and 101b shown in drawing 1 and the database-management-system (DBMS) server 103 is explained.
[0038] First, the configuration of the client applications 101a and 101b is explained. The application program (AP) 105 of the client application 101 directs the procedure which accesses a database (DB). An application program (AP) 105 is described using the set of functions of the application program interface (Application Program Interface:API) which the database-management-system (DBMS) client 106 offers. The database-management-system (DBMS) client 106 is bound to the client application 105.
[0039] The object management section 107 of the database-management-system (DBMS) client 106 manages the object 108 which the client application 101 treats. An object 108 is held on the client object cache 109 of the object management section 107.
[0040] The object status management table 110 in the object management section 107 manages the condition of the object 108 on the client object cache 109. The entry of the object status management table 110 is matched with each one object 108 of every.
[0041] The dictionary definition Management Department 111 in the database-management-system (DBMS) client 106 manages the dictionary information which is the DS (type) of an object 108, and the assembly of a definition of the access approach. Dictionary definition information required for actuation of an application program 105 shall be bound to the client application 101.
[0042] the temporary OID Management Department 112 in the database-management-system (DBMS) client 106 identifies each object 108 generated with the client application 101 -- temporary -- issue of OID is managed. temporary -- OID is created based on the generation serial number 113 by the order of generation managed at the temporary OID Management Department 112.
[0043] In the communication management section 114 of the database-management-system (DBMS) client 106, the communication link with the database-management-system (DBMS) server 103 is managed, and the updating wording of a telegram 115 for making the object actuation in the client application 101 reflect in a database (DB) is managed.
[0044] Next, the configuration of the database-management-system (DBMS) server 103 is explained. The database-management-system (DBMS) server 103 consists of assemblies of the process for realizing each functions which database system offers, such as transaction management and object management.
[0045] The system control section 116 in the database-management-system (DBMS) server 103

supervises the process group which constitutes a database (DB) system, and controls the whole system.
[0046] The transaction management section 117 in the system control section 116 manages the transaction performed with each client application 101.
[0047] Server applications 118a and 118b are the processes corresponding to the client applications 101a and 101b one [at a time], receive the demand from the client application 101, and tell a demand to each module (the Management Department, control section) of the database-management-system (DBMS) server 103, respectively.
[0048] The object management section 119 in the database-management-system (DBMS) server 103 receives and processes the object actuation demand of server application 118. The server object management section 119 operates an object 121 on the object cache 120 of the database-management-system (DBMS) server 103, and manages the condition of this object 121 on the object management information table 122 so that it may correspond to the object 108 of the client application 101.
[0049] the server OID Management Department 123 in a database management system (DBMS) 103 can set to a database (DB) system -- formal -- OID is published, and it manages so that uniqueness may be guaranteed.
[0050] the client application 101 published the server OID conversion Management Department 124 in a database management system (DBMS) 103 -- temporary -- as formal as OID -- correspondence relation with OID is managed. Correspondence relation is managed using the OID translation table 125 in the server OID conversion Management Department 124. About the detail of the OID translation table 125, it mentions later using drawing 19 .
[0051] The storing control section 126 manages data I/O with the datastore 104 which stores object data.

[0052] A datastore 104 is the device in which the storage for the permanent condition of an object 121 is given.
[0053] The communication management section 107 manages the buffer which receives a message from the client application 101.
[0054] Next, mode of processing of the object management approach by this operation gestalt is explained using drawing 2 . Drawing 2 is a flow chart which shows the concept of the processing flow of the application program in the client application in the object management approach by 1 operation gestalt of this invention. In addition, about a detailed processing flow, it mentions later below using drawing 3 .
[0055] In this operation gestalt, two objects are created and processing of the program of object actuation of setting up a link among those objects is explained, for example.
[0056] In step 201, the database-management-system (DBMS) client 106 calls the application program interface (API) function of the database-management-system (DBMS) client 106 for making database (DB) connection, and establishes the communication link with the client application 101 and the database-management-system (DBMS) server 103. About the detail of step 201, it mentions later using drawing 3 .
[0057] In step 202, the database-management-system (DBMS) client 106 calls the API function which starts a transaction. About the detail of step 202, it mentions later using drawing 6 . Then, adjustment is guaranteed by the database (DB) system about processing of a between from transaction initiation to the next transaction commitment.
[0058] In step 203, the database-management-system (DBMS) client 106 calls the API function which generates an object 108, and generates "an object 1." About the detail of step 203, it mentions later using drawing 7 and 8.
[0059] Here, there shall be a means to specify the type of the object to generate as an input of the object generation API. For example, the type identifier defined as the dictionary definition Management Department 111 shall be specified. A type identifier shall be given by general dictionary definition actuation of a database (DB) system, and the object of the DS based on the type definition corresponding to the specified type identifier shall be generated.
[0060] In step 204, the database-management-system (DBMS) client 106 generates "an object 2" like

step 203. Here, even if the type specified by "the object 1" and "an object 2" is the same, they may differ. However, the bidirectional relation of 1 to 1 shall be defined between each type.
[0061] In step 205, the database-management-system (DBMS) client 106 calls an object link setting API function, and sets up a link between "an object 1" and "an object 2." About the detail of step 205, it mentions later using drawing 9 , and 10 and 11. Here, the related identifier of the relation defined as the dictionary shall be specified as an input of the link setup API. By this assignment, the link between objects shall be set up according to the definition information on relation.
[0062] In step 206, the database-management-system (DBMS) client 106 calls the API function which carries out a transaction commitment. About the detail of step 206, it mentions later using drawing 12 . Thereby, the processing after step 203 is made to reflect in a database (DB).
[0063] In step 207, the API function of which database (DB) connection is canceled is called, and the communication link with the client application 101 and the database-management-system (DBMS) server 103 is canceled. An application program is ended after finishing step 207.
[0064] The detail of processing of the database (DB) connection API in the database-management-system (DBMS) client shown in step 201 of drawing 2 is explained using drawing 3 . Drawing 3 is a flow chart which shows the procedure of the database (DB) connection API in the object management approach by 1 operation gestalt of this invention.
[0065] In step 301, the object management section 107 of the database-management-system (DBMS) client 106 secures and initializes the data area of control information.
[0066] Here, the control data structure of the database-management-system (DBMS) client 106 is explained using drawing 4 . Drawing 4 is the explanatory view of the structure of the control data dealt with by the database-management-system (DBMS) client used in the object management approach by 1 operation gestalt of this invention.
[0067] by the configuration of this control information, it becomes a meaning with the client application 101 -- temporary -- issue of OID is managed and the information for performing an updating reflection demand collectively at the time of a transaction commitment is held.
[0068] As control information for the function of the database-management-system (DBMS) client 106, there are the client supervisory control header 401, the object management control header 402, a temporary OID supervisory control header 403, and a communication management control header 404. The address of such control information is held to each module of a database-management-system (DBMS) client as the client supervisory control header address at a global value, and each module accesses it at such control information.
[0069] The client supervisory control header 401 holds information (status) required in order to manage the whole process of the client application 101. For example, the condition (DB connection condition, transaction running state) of an application process is held.
[0070] The object management control header 402 holds the information which controls the object management section 107. The object management control header 402 holds the address of the cache control header 405, and the address of the object status management control header 406.
[0071] The cache control header 405 is used in order to manage the object cache 109, and it holds information required for management of the cash advance of the data of an object 108. The cache control header 405 holds the start address of the cache field of the object cache 109, and the start address of an available field.
[0072] The object status management table control header 406 manages the object status management table 110. In order to assign the entry of the object status management table 110, information, such as a start address of the table field of the object status management table 110 and a start address of an available table entry, is held.
[0073] the temporary OID supervisory control header 403 is temporary -- the information for publishing OID is managed. temporary -- the order serial number counter 113 of generation which becomes the origin of OID is held.
[0074] The communication management control header 404 is used in order to control communication management information, and the start address of the communication buffer 408 which is a field holding

communication link wording of a telegram etc. is held.
[0075] In the system of this operation gestalt, the reflection to the database-management-system (DBMS) server 103 of the object actuation performed with the client application 101 is put in block at the time of a commitment, and is performed. The contents of the demand to reflect are temporarily held as update information by the database-management-system (DBMS) client 106, and are pointed at from the object status management entry 407. since reflection can boil required updating demand informatior efficiently, reflection points at the required object status management entry 407 from the object status management control header 402, and each entry is connected with the pointer so that the object status management entry 407 for [further two or more] updating can be obtained. Here, this connection is called "the reflection demand chain 409."
[0076] Next, in step 302, the object management section 107 initializes the temporary OID Management Department 112, and the temporary OID Management Department 112 initializes the generation serial number 113.
[0077] In step 303, the object management section 107 initializes the communication management section 114, and equips the communication link demand to the subsequent database-management-system (DBMS) servers 103 with it.
[0078] In step 304, the database-management-system (DBMS) client 106 creates database (DB) connection-request wording of a telegram.
[0079] Here, the DS of the communication link wording of a telegram used in this operation gestalt is explained using drawing 5 . Drawing 5 is the explanatory view of the DS of the communication link wording of a telegram which client application creates in the object management approach by 1 operation gestalt of this invention.
[0080] The communication link wording of a telegram 501 consists of a wording-of-a-telegram control header 502 and a command parameter block 503 of plurality (n pieces). The wording-of-a-telegram control header 502 holds the information on the following required in order to control the communication link wording of a telegram 501.
[0081] (1) Formal" of "wording of a telegram (value which shows that it is the transmit data to a server)
(2) Size" of "wording of a telegram
(3) "head block offset" (the offset value which shows the location of the top command parameter block 503; given as an offset value from the head of the communication link wording of a telegram 501)
(-- 4) "a block total (n)" (total of the command parameter block 503 held to wording of a telegram 501)
The command parameter block 503 consists of block data 504 with the block control header 504. The block control header 504 has the information which controls each command parameter block 503. The block control header 504 holds a "block type", a "block size" (data length of a block), a "command identifier", "block data offset" (the offset value which shows the location of block data; offset value from the head of the command parameter block 503) (value which shows that it is a database (DB) connection request etc.), and "offset of degree block" (the offset value which shows the location of the following block; offset value from the head of the communication link wording of a telegram 501).
[0082] He sets the "block total" in the wording-of-a-telegram control header 502 to "1", and is trying to set up the value which shows "DB connection" to the "command identifier" in the block control header 504 of the first block 1 by using the structure explained above by the wording-of-a-telegram message of for example, a database (DB) connection request.
[0083] In step 305, the communication management section 114 transmits the communication link wording of a telegram 501 to the database-management-system (DBMS) server 103. After finishing step 305, database (DB) connection processing is ended.
[0084] Here, about the means and gestalt of interprocess communication, since the approach of this operation gestalt is mounted, a special function is not needed, but a well-known device may be used.
[0085] The detail of processing of the transaction initiation API in the database-management-system (DBMS) client shown in step 202 of drawing 2 is explained using drawing 6 . Drawing 6 is a flow chart which shows the procedure of the transaction initiation API function call in the object management approach by 1 operation gestalt of this invention.

[0086] In step 601, the database-management-system (DBMS) client 106 carries out a reset demand at the object management section 107, and a next object actuation demand is equipped with it by what (it is return to an initial state about control information) the object management section 107 resets.
[0087] In step 602, when a reset demand is carried out at the temporary OID Management Department 112 and the temporary OID Management Department 112 resets, temporary OID issue of a next object generate time is equipped with the object management section 107.
[0088] In step 603, the database-management-system (DBMS) client 106 creates the communication link wording of a telegram of a transaction initiation demand. Communication link wording of a telegram is created by the communication buffer field of the communication management section 114.
[0089] In step 604, the communication management section 114 transmits the created communication link wording of a telegram to the database-management-system (DBMS) server 103. A transaction initiation demand is told to the transaction management section 117, the transaction in the process of the client application 101 is recognized, and the database-management-system (DBMS) server 103 serves as an administration object.
[0090] This processing is ended after finishing step 604.
[0091] Next, the detail of processing of the object generation API in the database-management-system (DBMS) client shown in step 203 of drawing 2 is explained using drawing 7 and drawing 8 . Drawing 7 is a flow chart which shows the procedure of the object generation API in the object management approach by 1 operation gestalt of this invention, and drawing 8 is the explanatory view of the DS of the object generated.
[0092] Refer to the type definition information defined as the dictionary for the object management section 107 in step 701 shown in drawing 7 . The candidate for a type definition to refer to is specified with the parameter of the object generation API from the application program 105, and acquires the type definition information corresponding to this type identifier from the dictionary definition Management Department 111.
[0093] In step 702, the object management section 107 acquires a data area on the object cache 109. The magnitude of a data area computes required size with reference to the type definition information acquired at step 701.
[0094] In step 703, the object management section 107 initializes an object 108 according to the type definition information acquired at step 701. There is structure of holding link place information in the DS of an object 108, and link place information is initialized in this step.
[0095] In step 704, the object management section 107 assigns the entry of the object status management table 110, in order to manage the condition of this object.
[0096] In step 705, the object management section 107 initializes the entry of the object status management table 110 acquired at step 704. Here, it is set as ON which shows that it is in a new condition about the flag of the field holding the object condition in the entry of the object status management table 110.
[0097] In step 706, the object management section 107 connects the entry of this object status management table 110 with the reflection demand chain 409. The reflection demand chain 409 directs the following entry with a pointer.
[0098] After finishing step 706, object generation API processing is ended.
[0099] Next, use drawing 8 . The DS of the object 108 in the database-management-system (DBMS) client 106 is explained. The object 108 consists of an object control header 801 and an object data area 802. The object control header 801 consists of the following information, in order to carry out control information of the object data.
[0100] (-- 1) "the address of the object status management entry 806 of the object status management table 110"
(2) Classification" of "object
(3) Data size" of the whole "object
(4) "its own OID" (OID assigned to this object)
When generated by the client application 101, "the its own object identifier (OID)" is not set up.

temporary -- in the object actuation concerned, OID is published when an object identifier (OID) is needed (for example, when performing a link setup), and it is assigned. temporary [at the time of a link setup] -- about allotment of OID, it mentions later using drawing 9 .
[0101] The inside of "its own OID" consists of "a format of an object identifier (OID)" (value which shows whether it is "temporary" or it is "formal"), and a "discernment value" for uniqueness.
[0102] temporary -- in OID803, the value which shows that it is a temporary object identifier (OID) format in a "OID format" is set up, and the generation serial number of the generate time of the object for an assignment is set to a "discernment value."
[0103] formal -- in OID804, the value which shows that it is a formal object identifier (OID) format in a "OID format" is set up, and the value created so that it might become a meaning within a database (DB) system is set to a "discernment value" at the OID conversion Management Department 124 of the database-management-system (DBMS) server 103. Here, the permanent identifier PID based on object data storage positional information is used as a discernment value.
[0104] In the object data area 802, the "object data" in which the contents of the object are shown is held. The link place information 805 is also held to this field. "The link place OID" (object identifier of a link place object (OID)) is held at the link place information 805. temporary, if it becomes before a link place object is generated and carries out a transaction commitment -- OID803 will be held. formal, if a link place object is the existing object stored in the datastore -- OID804 will be held.
[0105] The object status management entry 806 of the object status management table 110 holds the following information, in order to manage the condition of an object 108.
[0106] (1) Object identifier (OID)" of "administration object
(2) "object status flag" (the value which shows that it is in the condition of generation/updating is held.) It turns on a flag, in being new.
(3) Address" of "object data (the start address of the object data area of a managed object; the column of the classification of the object control header 801 is made into the start address)
(4) Address" (address of the partial update information demanded from the managed object) of "partial update information
(5) Address" (address of the OID conversion information demanded from the managed object) of "object identifier (OID) conversion information
(6) "object identifier (OID) translation table registration request flag"
(-- 7) "the address of the continuation element entry of the updating chain 409"
Next, the detail of processing of the object link setup API in the database-management-system (DBMS) client shown in step 205 of drawing 2 is explained using drawing 9 , and 10 and 11. Drawing 9 is a flow chart which shows the procedure of the object link setup API in the object management approach by 1 operation gestalt of this invention.
[0107] Refer to the object related definition information defined as the dictionary definition Management Department 111 for the object management section 107 in step 901. The related target definition is specified by the related identifier as a parameter of the link setup API from an application program 105, and acquires the related definition information corresponding to this related identifier from the dictionary definition Management Department 111.
[0108] in step 902, the object management section 107 has a new linking agency object, and it is temporary -- it judges whether OID is assigned. Whether it is new judges with the "status flag" in the object status management entry 806 shown in drawing 8 . The "status flag" is turned on in being new. It is "its own OID" in the temporary object control header 801 shown in drawing 8 whether OID is assigned or not, and it judges by referring to a "OID format." temporary, when these values are not set up, since the value which shows that it is a temporary OID format in a "OID format", or the value which shows that it is a formal OID format is set up -- it judges with OID not being assigned.
[0109] temporary -- it progresses to step 903 and temporary, when OID is not assigned -- when OID is assigned, it progresses to step 906.
[0110] in step 903, the temporary OID Management Department 112 is new -- temporary -- OID is published. At this time, the temporary OID Management Department 112 increments the generation

serial number 113.
[0111] next, in step 904, the object management section 107 was published in the object control header 801 of a linking agency object, and the object status management entry 806 -- temporary -- OID is set up. that is, temporary to its own [of the object control header 801] OID -- OID803 is set up. The generation serial number which incremented at step 903 is set to the generation serial number 113. moreover, temporary to the "administration object OID" of the object status management entry 806 -- OID is set up.
[0112] Next, in step 905, the object management section 107 sets the "OID translation table registration request flag" in the object status management entry 806. When performing reflection processing by the database-management-system (DBMS) server 103 according to the value of this flag, entry registration to the OID translation table 125 can be efficiently performed by registering the entry about this object into the OID translation table 125. It progresses to step 906, after finishing step 905.
[0113] in step 909, an object identifier (OID) is judged like step 902 to the step 905 about a link place object from step 906, and temporary -- processing which assigns OID is performed.
[0114] that is, in step 906, the object management section 107 has a new link place object, and it is temporary -- it judges whether OID is assigned. temporary -- it progresses to step 907 and temporary, when OID is not assigned -- when OID is assigned, it progresses to step 910.
[0115] in step 907, the temporary OID Management Department 112 is newly temporary -- OID is published. At this time, the temporary OID Management Department 112 increments the generation serial number 113.
[0116] next, in step 908, it published in the object control header 801 of a link place object, and the object status management entry 806 -- temporary -- OID is set up.
[0117] Next, in step 909, the object management section 107 sets the "OID translation table registration request flag" in the object status management entry 806. It progresses to step 910, after finishing step 909.
[0118] In step 910, the object management section 107 creates the renewal demand information of a part for recording updating performed to a linking agency object by the processing concerned. About the detail of this updating demand information creation processing, it mentions later using drawing 10 and 11.
[0119] Furthermore, in step 911, the object management section 107 creates the updating demand information for recording updating performed to a link place object by the processing concerned like step 910. Also about the detail of this updating demand information creation processing, it mentions later using drawing 10 and 11.
[0120] After finishing step 911, link setting API processing is ended.
[0121] Next, the detail of processing of the update information creation in the database-management-system (DBMS) client shown in step 910,911 of drawing 9 is explained using drawing 10 and 11. Drawing 10 is a flow chart which shows the procedure of the update information creation at the time of a link setup in the object management approach by 1 operation gestalt of this invention, and drawing 11 is the explanatory view of the DS treated by the update information creation processing at the time of a link setup in the object management approach by 1 operation gestalt of this invention.
[0122] **.
[0123] Refer to the related definition information acquired from nothing [dictionary definition management / 111] for the object management section 107 in step 1001. From the acquired related definition information, in an object, offset of the location holding link place information is acquired, and link place information is embedded in a linking agency object. Here, link place information is OID of a link place object.
[0124] That is, in drawing 11 , if object 1101a is made into a link place object and object 1101b is made into a linking agency object, "temporary OIDa" of link place object 1101a will be embedded to the link place information 1102 on linking agency object 110b.
[0125] In step 1002, the object management section 107 acquires an object status management entry from a linking agency object with reference to the address currently held to the "object control header."

[0126] That is, in drawing 11 , object status management entry 1104b of the object status management table 110 is acquired with reference to the address currently held at the head of the "object control header" of linking agency object 1101b.
[0127] in step 1003, the object management section 107 has temporary OID of a link place object -- it judges whether it is OID. This judgment is performed with reference to the "OID format" in OID. temporary -- in OID, it progresses to step 1004, and it is temporary -- when it is not OID, it progresses to step 1006.
[0128] That is, in drawing 11 , it judges with reference to the "OID format" (refer to drawing 8) of "its own OID" in the "object control header" of link place object 1101a. here, both link place object 1101a and linking agency object 1101b are temporary -- since it is OID, it shall progress to step 1004
[0129] In step 1004, the object management section 107 creates OID conversion information based on offset of the location (link place information in a linking agency object) which embeds OID of a link place object. the offset set as this OID conversion information has the temporary database-management-system (DBMS) server 103 -- it is used in order to obtain efficiently the location where OID was embedded.
[0130] That is, in drawing 11 , update position offset is set to the OID conversion information 1105. This update position offset is offset of the link place information 11102 on linking agency object 1101b.
[0131] Next, in step 1005, the object management section 107 sets the address of OID conversion information as the object management management entry of a linking agency object. plurality is temporary to one object -- when OID is embedded, in order to cope with it, OID conversion information is connected by the address pointer, and the address of the element (OID conversion information 1105) of the head of a chain of OID conversion information is held to an object status management entry.
[0132] That is, in drawing 11 , the address of the OID conversion information 1105 is set as object management management entry 1104of linking agency object 1101b b. The setting location of the address is a location in front of an OID conversion request flag, as explained in drawing 8 . plurality is temporary -- when OID is embedded, as shown in drawing 11 , the following OID conversion information is connected by the address pointer of the front OID conversion information 1105.
[0133] In step 1006, the object management section 107 creates the partial update information for setting up link place information. Partial update information consists of "update position offset", "updating range", and "a start address of the contents of updating data." In order to cope with two or more renewal of a part which receives one object, partial update information is connected by the "address pointer." In an object status management entry, the address of the head element of a partial update information chain is held.
[0134] That is, in drawing 11 , the partial update information 1103 consists of "update position offset", "updating range size", "a start address of the contents of updating data", and "a following address pointer of partial update information." "The start address of the contents of updating data" is a start address of the link place information 1102. Object status management entry 1104b holds the address of the element (partial update information 1103) of the head of a chain of partial update information. The location is a location in front of the address of the OID conversion information 1105, as explained in drawing 8 .
[0135] In step 1007, the object management section 107 sets the address of partial update information as the object status management entry of a linking agency object. When partial update information is created, the newly created partial update information is already connected with the chain of created partial update information.
[0136] That is, in drawing 11 , the address of the partial update information 1103 is set as the location in front of [of the "OID conversion request flag" of object status management entry 1104b] two.
[0137] In step 1008, the object management section 107 sets up a linking agency object status management entry. That is, the value which shows "renewal" of the "status flag" of a linking agency object status management entry is turned ON. When creating reflection information wording of a telegram by next transaction commitment, since [which carries out a thing judging] reflection of renewal of a part is demanded of the object concerned, this flag is used.
[0138] That is, in drawing 11 , the value which shows "renewal" of the object status flag of object status

management entry 1104b of linking agency object 1101b is turned ON.
[0139] In step 1009, the object management section 107 connects a linking agency object status management entry with a reflection demand chain.
[0140] That is, in drawing 11 , object status management entry 1104of linking agency object 1101b b is connected with the reflection demand chain 409.
[0141] After finishing step 1009, update information creation processing is ended.
[0142] As a result of performing update information creation processing in which it explained in drawing 10 in drawing 11 here, OID (temporary OIDa) assigned to link place object 1101a is set as the link place information 1102 in the object data area of linking agency object 1101b. In order to record this setting demand, the partial update information 1103 is assigned. The partial update information 1103 is connected with the partial update information chain of object status management entry 1104b.
[0143] moreover, OID of link place object 1101a is temporary -- when it is OID, the OID conversion information 1105 is created and it is tied to the OID conversion information chain of object status management entry 1104b. a link place is formal -- when it has OID, about this link setup, OID conversion information becomes unnecessary.
[0144] In addition, in the example mentioned above, since it is considering as bidirectional relation, the same structure has been taken also about the link place object.
[0145] Next, the detail of the processing of API in the database-management-system (DBMS) client shown in step 206 of drawing 2 which carries out a transaction commitment is explained using drawing 12 and drawing 13 . Drawing 12 is a flow chart which shows the procedure of the transaction commitment API in the object management approach by 1 operation gestalt of this invention, and drawing 13 is the explanatory view of the communication link wording-of-a-telegram DS of the transaction commitment demand in the object management approach by 1 operation gestalt of this invention.
[0146] In step 1201, the database-management-system (DBMS) client 106 acquires the top element address of a reflection demand chain of an object status management entry from an object control header.
[0147] That is, in drawing 11 , object status management entry 1104b is acquired from the object control header of object 1101b, and the start address of the reflection demand chain 409 is acquired from object status management entry 1104b.
[0148] In step 1202, the database-management-system (DBMS) client 106 judges whether an element is in the reflection demand chain acquired at step 1201, i.e., is the reflection demand made?. When the reflection demand is not made, it progresses to step 1203, and when the reflection demand is made, it progresses to step 1205.
[0149] Here, it explains to step 1205 spontaneously that the reflection demand is made since link place object 1101a and linking agency object 1101b are linked.
[0150] In step 1205, the database-management-system (DBMS) client 106 sets the generation serial number of the temporary OID Management Department as communication link wording of a telegram as an OID conversion total. This value tells the total which needs OID conversion by the server, and since the field of an OID translation table is initialized, it is used.
[0151] here, temporary to two objects of Objects 1101a and 1101b -- OID is published, and considering the case where package transmission of these two objects is carried out, "2" is set to the conversion OID total 302 of the communication link wording of a telegram 1301 of the commitment demand shown in drawing 13 .
[0152] In step 1206, the database-management-system (DBMS) client 106 follows the reflection demand chain 409, and acquires the address of object status management entry 1104b to be reflected.
[0153] In step 1207, the database-management-system (DBMS) client 106 judges whether the value which shows "it is new" with the "status flag" in the object status management entry acquired at step 1206 is ON. When a "new flag" is ON, it progresses to step 1208 and object generation demand wording of a telegram is created. Object generation demand wording of a telegram is later mentioned using drawing 14 about the detail of this object generation demand wording-of-a-telegram creation, although it

is created as shown in the command parameter 1303 of drawing 13 . When a "new flag" is not ON, it progresses to step 1209.
[0154] In step 1209, the database-management-system (DBMS) client 106 judges whether there is any continuation of a reflection demand chain. When there is a continuation, the wording of a telegram about all generation demands is created by repeating processing from return and step 1206 to step 1209 to step 1206. In the 2nd step 1208, the object generation demand wording of a telegram shown in the command parameter 1304 of drawing 13 is created. At step 1209, when there is no continuation, it progresses to step 1210.
[0155] In step 1210, the database-management-system (DBMS) client 106 returns to the head of a reflection demand chain. That is, with reference to object control header 1101b shown in drawing 11 , the address of object status management entry 1104b of the object status management table 110 is acquired.
[0156] In step 1211, the database-management-system (DBMS) client 106 follows a reflection demand chain, and acquires the address of an object status management entry to be reflected. That is, the reflection demand chain 409 shown in drawing 11 is followed, and the address of the following object status management entry is acquired.
[0157] In step 1212, the database-management-system (DBMS) client 106 judges whether the value which shows updating with the status flag in an object status management entry is ON. When an update flag is ON, it progresses to step 1213 and the renewal demand wording of a telegram of an object is created. The renewal demand wording of a telegram of an object is later mentioned using drawing 15 about the detail of this renewal demand wording-of-a-telegram creation of an object, although it is created as shown in the command parameter 1305 of drawing 13 . When an update flag is not ON, it progresses to step 1214.
[0158] In step 1214, the database-management-system (DBMS) client 106 judges whether there is any continuation of a reflection demand chain. When there is a continuation, the wording of a telegram about all updating demands is created by repeating processing from return and step 1211 to step 1214 to step 1211. In the 2nd step 1213, the renewal demand wording of a telegram of an object shown in the command parameter 1308 of drawing 13 is created. At step 1214, when there is no continuation, it progresses to step 1215. Thereby, the wording of a telegram about all updating demands is created.
[0159] In step 1215, the database-management-system (DBMS) client 106 creates the wording of a telegram of a transaction commitment demand. The database-management-system (DBMS) client 106 secures the wording-of-a-telegram field of a commitment demand to the communication management section 114, and sets the command identifier which shows that it is a commitment demand as the command parameter block 1311 of the transaction commitment demand wording of a telegram shown in drawing 13 . In addition, when there is degree block, offset of degree the block is set to the block [degree] offset behind a command identifier.
[0160] Next, in step 1216, from the communication management section 114, the database-management-system (DBMS) client 106 transmits wording of a telegram to a server 103, and ends transaction commitment demand processing.
[0161] In addition, in step 1202, when it progresses to step 1203 noting that there is no reflection demand, like step 1215, the wording of a telegram of a transaction commitment demand is created, and in step 1204, the database-management-system (DBMS) client 106 transmits wording of a telegram to a server 103, and ends transaction commitment demand processing from the communication management section 114 further at step 1203.
[0162] Next, the DS of transaction commitment demand wording of a telegram is explained using drawing 13 . The wording-of-a-telegram control header of the communication link wording of a telegram 1301 (1) "the format of wording of a telegram" (value which shows that it is the transmit data to a server), (2) Size" of "wording of a telegram, (3) "head block offset" (the offset value which shows the location of the top command parameter block 1303; given as an offset value from the head of the communication link wording of a telegram 1301), (4) It consists of "block total" (total of the command parameter block 1303 held to the communication link wording of a telegram 1301), and (5) "a

conversion OID total" (it sets up at step 1205 of drawing 12). From the data set as the "block total", it is shown that the communication link wording of a telegram 1301 is constituted by five command parameter blocks 1303, 1304, 1305, 1308, and 1311. Moreover, it is shown by "the conversion OID total 1302" that the number of the elements registered into a conversion OID table by this communication link wording-of-a-telegram processing is "2."

[0163] "Generation" is set as the "command identifier" by the 1st command parameter block 1303, "Object a" is set to it as "object data", and the generation demand of Object a is describing. In the command parameter block 1303, the "OID translation table registration request flag" is set as ON, and when processing a demand by the database-management-system (DBMS) server 103, registering the entry about this object into the OID translation table 125 is directed. Moreover, the head location of the command parameter block 1304 is directed by "block [degree] offset."

[0164] At the 2nd command parameter block 1304, the generation demand about Object b is describing like the command parameter block 1303. The "OID translation table registration request flag" of the command parameter block 1304 is set as ON, and the head location of the command parameter block 1305 is directed by "block [degree] offset."

[0165] "Updating" is set as the "command identifier" by the 3rd command parameter block 1305, "temporary OIDa" is set to it as "data for updating", and the updating demand of Object a is describing. By "block [degree] offset", the head location of the command parameter block 1308 is directed. The number of the chains of the partial update information 1103 shown in drawing 11 is set to "the number of renewal of a part." "Partial update information offset" is directing the head location of the partial update information 1306. The number of the chains of the OID conversion information 1105 which showed "the number of OID conversion" in drawing 11 is set up. "OID conversion information offset" is directing the head location of the OID conversion information 1307.

[0166] Furthermore, the setting information on the link place information on Object b is describing at the partial update information 1306. The contents of the partial update information 1103 which showed "renewal offset of a part", "renewal offset of a part", and the "contents of updating data" (link place information) to drawing 11 are set up.

[0167] moreover, to the OID conversion information 1307, the object b within link place information is temporary -- the information for changing OID is describing. The contents of the OID conversion information 1105 which showed "OID conversion location offset" in drawing 11 are set up.

[0168] To the 4th command parameter block 1308, the partial update information 1309, and the OID conversion information 1310, the updating demand of Object b is describing like the command parameter block 1305, the partial update information 1306, and the OID conversion information 1307.

[0169] The commitment demand is describing at the 5th command parameter block 1311. It is guaranteed by this command parameter block 1311 by performing a transaction commitment by the database-management-system (DBMS) server 103 that the demand by the command parameter block 1310 is reflected from the command parameter block 1303 to a datastore.

[0170] Next, the detail of processing of the object generation demand wording-of-a-telegram creation in the database-management-system (DBMS) client shown in step 1208 of drawing 12 is explained using drawing 14 . Drawing 14 is a flow chart which shows the procedure of the object generation demand wording-of-a-telegram creation in the object management approach by 1 operation gestalt of this invention.

[0171] In step 1401, the database-management-system (DBMS) client 106 computes an area size required for wording of a telegram based on the data size of an object 108.

[0172] In step 1402, the communication management section 114 secures a wording-of-a-telegram field based on the value computed at step 1401.

[0173] In step 1403, the database-management-system (DBMS) client 106 judges whether the "OID translation table registration request flag" in object status management entry 1104b is ON. When a "OID translation table registration request flag" is ON, it progresses to step 1404, and in not being ON, it progresses to step 1405.

[0174] In step 1404, the database-management-system (DBMS) client 106 sets the "OID translation

table registration request flag" in a command parameter as ON. That is, the command parameter block 1303 shown in drawing 13 and the "OID translation table registration request flag" in 1304 are set as ON.

[0175] In step 1405, the database-management-system (DBMS) client 106 copies the data of an object 108 to the new object data area in a command parameter. That is, object data are set as the field of the "object data" in the command parameter blocks 1303 and 1304. After finishing step 1405, processing of generation demand wording-of-a-telegram creation is ended.

[0176] Next, the detail of processing of the renewal demand wording-of-a-telegram creation of an object in the database-management-system (DBMS) client shown in step 1213 of drawing 12 is explained using drawing 15 . Drawing 15 is a flow chart which shows the procedure of the renewal demand wording-of-a-telegram creation of an object in the object management approach by 1 operation gestalt of this invention.

[0177] In step 1501, the database-management-system (DBMS) client 106 follows the chain of the partial update information of an object status management entry, counts the number of updating demands, and computes size required for the command parameter block of wording of a telegram. That is, the chain of the partial update information 1103 shown in drawing 11 is followed, and the number of entries of partial update information is counted.

[0178] In step 1502, the database-management-system (DBMS) client 106 counts the temporary OID number which follows and changes an OID conversion information chain, and computes size required for the command parameter block of wording of a telegram. That is, the chain of the OID conversion information 1105 shown in drawing 11 is followed, and the number of conversion demands of OID is counted.

[0179] In step 1503, the database-management-system (DBMS) client 106 computes size required of the whole updating demand command parameter block based on the value acquired at step 1501 and step 1502, and the communication management section 114 secures a wording-of-a-telegram field.

[0180] In step 1504, the database-management-system (DBMS) client 106 sets OID of the object for updating as communication link wording of a telegram. That is, temporary OIDb of object status management entry 1104b shown in drawing 11 is set as "the candidate for updating" of the command parameter block 1305 shown in drawing 13 .

[0181] In step 1505, the database-management-system (DBMS) client 106 follows the chain of the partial update information 1103, and acquires one partial update information 1103.

[0182] In step 1506, with reference to the partial update information acquired at step 1505, the database-management-system (DBMS) client 106 acquires "partial update position offset" and the "updating range", and sets this as communication link wording of a telegram. Moreover, with reference to the contents of updating of the object on an object cache, the contents of updating data are set as wording of a telegram. That is, "partial update position offset" of the partial update information 1103 shown in drawing 11 and "the renewal range size of a part" are acquired, and it is set as "partial update position offset" of the partial update information 1306 shown in drawing 13 , and "the renewal range size of a part", and the contents of updating data (link place information) are set up.

[0183] In step 1507, it judges whether it continues by partial update information chain, and there is any element by the database-management-system (DBMS) client 106. When there is a continuation, processing of return and step 1505 to the step 1507 is repeated to step 1505, and the wording of a telegram of all partial update information is created. That is, the partial update information 1306 and the partial update information 1309 which were shown in drawing 13 are created.

[0184] On the other hand, in the judgment of step 1507, when there is no continuation, it progresses to step 1508.

[0185] In step 1508, the database-management-system (DBMS) client 106 sets the offset information on a location that update information was set up in the number of update information, and wording of a telegram as the block control header of a command parameter block. That is, the offset information on a location that set the number of update information as the "number of renewal of a part" of the command parameter block 1305 shown in drawing 13 , and update information was set as it in wording of a

telegram at "partial update information offset" is set up.
[0186] In step 1509, the database-management-system (DBMS) client 106 judges whether OID conversion information is in updating data. That is, the "OID conversion information address" of object status management entry 1104b shown in drawing 11 judges whether the start address of the OID conversion information 1105 is held. It judges whether in this object, OID conversion is required.
[0187] When there is OID conversion information, it progresses to step 1510, and when there is no OID conversion information, it progresses to step 1512.
[0188] In step 1510, the database-management-system (DBMS) client 106 follows an OID conversion information chain, and acquires one OID conversion information. That is, the OID conversion information 1105 shown in drawing 11 is acquired.
[0189] Next, in step 1511, the database-management-system (DBMS) client 106 sets OID conversion information as wording of a telegram. Here, the offset which shows the location which carries out OID conversion is set up.
[0190] That is, the OID conversion information 1307 shown in drawing 13 is set up. Moreover, "OID [degree] conversion offset" is set up at this time.
[0191] In step 1512, the database-management-system (DBMS) client 106 judges whether it continues by OID conversion information chain, and there is any element. When there is a continuation, processing of return and step 1510 to the step 1512 is repeated to step 1510. Thereby, the wording of a telegram of all OID conversion information is created. That is, the OID conversion information 1307 and 1310 is set up. When there is no continuation, it progresses to step 1513.
[0192] In step 1513, the database-management-system (DBMS) client 106 sets the offset which shows the location of the number of the OID conversion information created by processing of step 1509 to the step 1512, and the OID conversion information on the inside of wording of a telegram as the block control header of a command parameter block. That is, the number of OID conversion information is set as the "number of OID conversion" of the command parameter blocks 1305 and 1308 shown in drawing 13 , and the offset which shows the location of the OID conversion information on the inside of wording of a telegram to "OID conversion information offset" is set up.
[0193] After finishing step 1513, the updating demand wording-of-a-telegram creation processing concerned is ended.
[0194] Next, the detail of processing of the transaction commitment in the database-management-system (DBMS) server shown in step 206 of drawing 2 is explained using drawing 16 . Drawing 16 is a flow chart which shows the procedure of the transaction commitment in the object management approach by 1 operation gestalt of this invention.
[0195] In step 1601, server application 118a acquires the address of wording of a telegram from the communication management section 107.
[0196] In step 1602, 124 secures the field which can hold the element array for a total based on the conversion OID total in communication link wording of a telegram (conversion OID total 1302 shown in drawing 13), and initializes the OID translation table 125.
[0197] In step 1603, server application 118 acquires the command parameter block 1303 from the communication link wording of a telegram obtained at step 1601 with reference to "head block offset" of a wording-of-a-telegram control header.
[0198] In step 1604, it judges whether server application 118 is a value the "command identifier" of the command parameter block 1303 indicates "generation" to be. In "generation", it progresses to step 1605. In step 1605, the object management section 119 performs wording-of-a-telegram analysis processing of a generation demand. About the detail of wording-of-a-telegram analysis processing of a generation demand, it mentions later using drawing 17 . It progresses to step 1610, after finishing step 1605.
[0199] The following command parameter block is acquired in step 1610. Since the "command identifier" of the command parameter block 1303 is "generation" and the "command identifier" of the continuing command parameter block 1304 is "generation" as shown in drawing 13 , generation command processing by step 1605 is performed.
[0200] On the other hand, when it is not "generation" in step 1604, it goes to step 1606 and a command

identifier judges whether it is the value which shows "updating."
[0201] Since the "command identifier" of the command parameter blocks 1305 and 1308 shown in drawing 13 is "updating", it progresses to step 1607 and the object management section 119 performs wording-of-a-telegram analysis processing of an updating demand. About the detail of wording-of-a-telegram analysis processing of an updating demand, it mentions later using drawing 18 . It progresses to step 1610, after finishing step 1607.
[0202] On the other hand, when it is not "updating" at step 1606, it goes to step 1608 and a command identifier judges whether it is the value which shows "a commitment."
[0203] Since the "command identifier" of the command parameter block 1311 shown in drawing 13 is "a commitment", it progresses to step 1609 and server application 118 requires a transaction commitment of the transaction management section 117. After finishing step 1609, the commitment processing concerned is ended.
[0204] On the other hand, when demand classification is not "a commitment" in step 1608, it progresses to step 1610, and at step 1610, the following command parameter block in wording of a telegram is acquired, and it returns to step 1604. Processing to all "generation" to a "commitment" demand and the reflection demand of "updating" is performed by the above repeat.
[0205] Next, the detail of object generation command processing in the database-management-system (DBMS) server shown in step 1605 of drawing 16 is explained using drawing 17 and drawing 19 . Drawing 17 is a flow chart which shows the procedure of the generation command in the object management approach by 1 operation gestalt of this invention, and drawing 19 is the explanatory view of the structure of the object data within the DBMS server in the object management approach by 1 operation gestalt of this invention.
[0206] In step 1701, the object management section 119 acquires the object data size in a command parameter block, and it secures an object cache field so that the object of object data size can be held.
[0207] That is, the configuration of the "object data" of the command parameter blocks 1303 and 1304 shown in drawing 13 has composition as shown in drawing 8 , and acquires object data size from "the size of the whole object" in the object control header 801 of the object data 108 shown in drawing 8 .
[0208] In step 1702, the object management section 119 is copied to the field which secured the object data in a command parameter block at step 1701. That is, it copies to object 1901a of the object cache 120 which showed the "object data" of the command parameter blocks 1303 and 1304 shown in drawing 13 to drawing 19 .
[0209] In step 1703, the object management section 119 requires object storing of the storing control section 126. a permanent identifier (PID) is given and more formal to this object than the OID Management Department 124, when a storing location is decided -- OID is assigned. here, it was assigned -- formal -- OID is set as the retention area (field of formal OIDa in drawing 19) of its own [of the control header of object 1901a on the object cache 120] OID. That is, "formal OIDa" is set as "its own OID" of object 1901a on the object cache 120 of drawing 19 , and "formal OIDb" is set as "its own OID" of object 1901b.
[0210] In step 1704, the object management section 119 assigns and initializes an object status management entry to this object data. Thereby, this object will be in an activated state.
[0211] In step 1705, the object management section 119 judges whether the command parameter block 1303 and the "OID translation table registration request flag" in 1304 are ON. When a flag is ON, it progresses to step 1706.
[0212] In step 1706, the object management section 119 is registered into an OID translation table based on the serial number in temporary OID of an object. That is, the structure in the command parameter blocks 1303 and 1304 shown in drawing 13 is as having been shown in drawing 8 , based on the generation serial number in temporary OID803 in "its own OID" in the object control header 801 of an object 108 shown in drawing 8 , in the OID translation table 125 shown in drawing 19 , decides the registration location of the entry about this object, and sets up the address of the object data activated.
[0213] When a flag is not ON at step 1705 after finishing step 1706 or, generation command processing is ended.

[0214] Next, the detail of renewal command processing of an object in the database-management-system (DBMS) server shown in step 1607 of drawing 16 is explained using drawing 18 . Drawing 18 is a flow chart which shows the procedure of the updating command in the object management approach by 1 operation gestalt of this invention.
[0215] In step 1801, the object management section 119 activates the object for updating into an object cache. Here, when the candidate for updating is a new object, an object object can be acquired by being already an activated state and referring to an OID translation table previously, by processing explained by drawing 17 . moreover, in the case of the existing object by which the candidate for updating is stored in the datastore, it is specified in communication link wording of a telegram -- formal -- OID shall be activated from a datastore using a well-known activation device
[0216] In step 1802, the object management section 119 acquires one partial update information under block with reference to a block control header. That is, the partial update information 1306 is acquired from "partial update information offset" of the command parameter block 1305 shown in drawing 13 .
[0217] In step 1803, the object management section 119 rewrites the object data acquired at step 1801 based on the partial update information acquired at step 1802. That is, in step 1801, partial update information is put into the initialized field.
[0218] In step 1804, the object management section 119 judges whether a continuation is in the partial update information processed at previous step 1803 with reference to the block control header of update information.
[0219] When there is a continuation, return and all renewal of a part are made to be performed to step 1802. That is, from "partial update information offset" of the command parameter block 1308 shown in drawing 13 , the partial update information 1309 is acquired and the object data acquired at step 1801 are rewritten based on the acquired partial update information. When there is no continuation, it is progress to step 1805.
[0220] In step 1805, the object management section 119 judges whether OID conversion information is held during a block with reference to a block control header. That is, it judges whether OID conversion information is held from "OID conversion information offset" of the command parameter block 1305 shown in drawing 13 . When progressing to step 1812 when not holding, and holding, it progresses to step 1806.
[0221] In step 1806, the object management section 119 acquires one OID conversion information out of a block. That is, the OID conversion information 1307 shown in drawing 13 is acquired.
[0222] in step 1807, the object management section 119 is embedded in object data based on the offset within the OID conversion information acquired at step 1806 -- temporary -- OID is acquired. That is, "temporary OIDa" "for updating" is acquired. [in the command parameter block 1305 shown in drawing 13]
[0223] in step 1808, the object management section 119 was acquired at step 1807 -- temporary -- temporary with reference to the contents of OID -- the generation serial number currently held at the in-house data of OID is acquired. that is, temporary -- as the structure of OID was shown in drawing 8 , it is temporary -- the "generation serial number" of OID803 is acquired. [of drawing 8]
[0224] In step 1809, the object management section 119 acquires the entry corresponding to an object for this value as an OID translation table array element location based on the value acquired at step 1808. As shown in drawing 19 , the address of the object activated on the object cache 120 is set to this entry. to the control header of an object 1901, he is formal -- formal by referring to this, since OID is set up -- OID is acquired.
[0225] in step 1810, the object management section 119 is [in object data] temporary -- OID was obtained at step 1809 -- formal -- it rewrites to OID. That is, formal OIDb is written in the "link place OID" in the object control header 801 shown in drawing 8 in object 1901a shown in drawing 19 .
[0226] In step 1811, the object management section 119 judges whether the OID conversion information on a continuation is during a block. When there is OID conversion information, processing of step 1806 to the step 1811 is repeated until it processes return and all OID conversion to step 1806. Thereby, formal OIDa is written in by the "link place OID" in object 1901b shown in drawing 19 .

[0227] On the other hand, in step 1811, when there is no OID conversion information, it progresses to step 1812.
[0228] In step 1812, the object management section 119 requires that old updating should be reflected in a datastore from the storing control section 126. After finishing step 1812, the updating command processing concerned is ended.
[0229] Here, the object DS in a database-management-system (DBMS) server is explained using drawing 19 .
[0230] Object a1901a and object b1901b are activated on the object cache 120. Each entry of Object a and Object b is registered into the OID translation table 125 of the OID Management Department 124. Here, Object a should be generated by the a-th by the generation serial number, and the entry is registered into the a-th in the array of the OID translation table 125 according to the order of generation. Similarly, the entry is registered into the b-th in the array of the OID translation table 125 also about Object b.
[0231] By step 1703 of drawing 17 , "formal OIDa" and "formal OIDb" are set as the maintenance field of "their own OID" by the object control header of object a1901a and object b1901b, respectively.
[0232] Moreover, "formal OIDb" and "formal OIDa" are set as "the link place OID" by step 1810 of drawing 18 , respectively for the link place information on object a1901a and object b1901b.
[0233] As mentioned above, although 1 operation gestalt of this invention was explained, this illustrates and this invention is not restricted by this.
[0234] In addition, in the example mentioned above, although the multiplicity of the relation between objects was set to 1 to 1, this invention is applicable also about many to many relation. As the many-to-many-related mounting approach, the link place OID is not held soon, but, generally, it is mounted through the collection function to manage the set of OID, and is interpreted as OID of the element of a collection being OID of a link place object. in this case, temporary, since OID is treated fundamentally in the above-mentioned method -- formal in OID -- mounting is possible if it changes into OID.
[0235] Moreover, in the example mentioned above, although the period unit which guarantees the uniqueness of a generation serial number is made into the transaction, setting up a unit suitably according to the gestalt of client applications, such as timing which communicates with a server, and database (DB) connection, a client process unit, is also considered.
[0236] according to this operation gestalt, in the object-oriented database system of a client-server configuration, it carried out based on the order serial number of generation to the new object -- temporary -- more formal using OID -- it becomes possible to process efficiently object actuation with client application which needs OID, without performing the communication link with a database-management-system (DBMS) server (reducing the count of a communication link).
[0237] moreover, plurality is temporary -- the count of a communication link is reducible by transmitting OID to a server from a client collectively.
[0238] moreover, temporary by adding OID conversion information to the wording of a telegram of a reflection demand -- formal from OID -- conversion to OID can be efficiently performed now.
[0239]
[Effect of the Invention] Object actuation with client application can be processed efficiently, without communicating with a server in the object-oriented database system which consists of client-server molds according to this invention, guaranteeing object epicritic within a system.

[Translation done.]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開平9－305468
(43)公開日　平成９年(1997)11月28日

(51)Int.Cl.⁶	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　12/00	５３３		Ｇ０６Ｆ　12/00	５３３Ｊ	
	５４７			５４７Ａ	
9/44	５３０		9/44	５３０Ｃ	
13/00	３５７		13/00	３５７Ｚ	

審査請求　未請求　請求項の数５　ＯＬ　（全 30 頁）

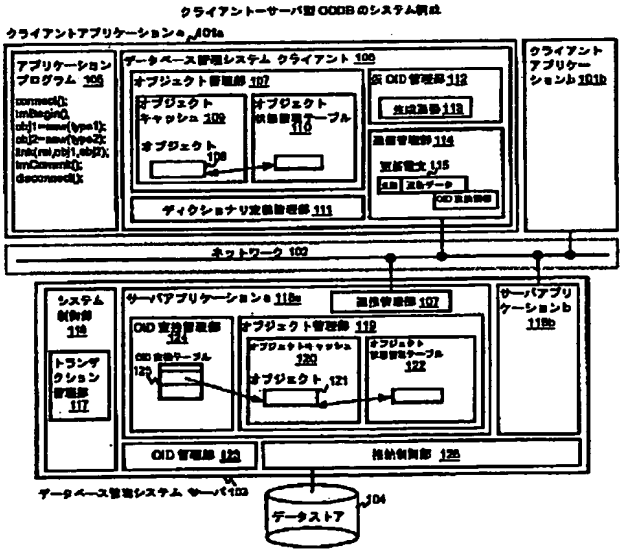
(21)出願番号　特願平8－120498

(22)出願日　平成８年(1996)５月15日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　小林　挙
神奈川県川崎市幸区鹿島田890番地の12
株式会社日立製作所情報・通信開発本部内
(72)発明者　山本　洋一
神奈川県川崎市幸区鹿島田890番地の12
株式会社日立製作所情報・通信開発本部内
(72)発明者　浅見　真人
神奈川県川崎市幸区鹿島田890番地の12
株式会社日立製作所情報・通信開発本部内
(74)代理人　弁理士　春日　譲

(54)【発明の名称】　オブジェクト管理方法

(57)【要約】
【課題】本発明の目的は、クライアントーサーバ型で構成されるオブジェクト指向データベースシステムにおいて、システム内でのオブジェクト識別性を保証しつつ、サーバと通信を行うことなく、クライアントアプリケーションでのオブジェクト操作を効率よく処理できるオブジェクト管理方法を提供するにある。
【解決手段】データベース管理システム（ＤＢＭＳ）クライアント１０６のオブジェクト管理部１０７は、オブジェクトの生成順の通し番号（生成通番１１３）を基にした仮のオブジェクト識別子（ＯＩＤ）を、生成したオブジェクト１０８に割り付ける。データベース管理システム（ＤＢＭＳ）クライアント１０６が作成する更新電文１１５には、この電文中に含む仮ＯＩＤの位置を示すＯＩＤ変換情報を付加する。データベース管理システム（ＤＢＭＳ）サーバ１０３は、生成したオブジェクトをデータストア１０４でアクセス効率が良くなるように格納し、ＯＩＤ変換情報に基づき、仮ＯＩＤをシステム内で一意となる正式なＯＩＤに変換する。



【特許請求の範囲】
【請求項１】　データベース管理システムクライアントは、クライアントアプリケーションで生成したオブジェクトを一時的な仮オブジェクト識別子で管理し、
上記オブジェクトについて反映要求をされたデータベース管理システムサーバは、上記仮オブジェクト識別子をデータベースシステムで一意となる正式オブジェクト識別子に変換して、上記オブジェクトのデータストアへの格納を制御することを特徴とするオブジェクト管理方法。
【請求項２】　請求項１記載のオブジェクト管理方法において、
上記データベース管理システムクライアントは、上記データベース管理システムサーバに対して上記オブジェクトについて反映要求をする際に、反映要求を指示する情報に、上記仮オブジェクト識別子から上記正式オブジェクト識別子に変換するための、補足的なオブジェクト識別子変換情報を付加することを特徴とするオブジェクト管理方法。
【請求項３】　請求項１記載のオブジェクト管理方法において、
上記データベース管理システムクライアントは、上記オブジェクトを生成順の番号で管理し、この生成順の番号に基づいて上記仮オブジェクト識別子をそれぞれのオブジェクトに割り付けることを特徴とするオブジェクト管理方法。
【請求項４】　請求項１記載のオブジェクト管理方法において、
上記データベース管理システムクライアントは、複数のオブジェクトを一括して上記データベース管理システムサーバに送信して、反映要求をすることを特徴とするオブジェクト管理方法。
【請求項５】　請求項４記載のオブジェクト管理方法において、
上記複数のオブジェクト間にリンクを設定し、上記反映要求を指示する情報に、リンク先情報を設定する部分更新情報を付加することを特徴とするオブジェクト管理方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、オブジェクト管理方法に係り、特に、クライアントーサーバ型で構成されるオブジェクト指向データベースシステムに用いるに好適なオブジェクト管理方法に関する。
【０００２】
【従来の技術】ネットワークを介したクライアントーサーバ型のシステム構成は、可用性や保守容易性を備えるための手段として、多くの計算機システムに適用されている。オブジェクト指向データベース（Ｏｂｊｅｃｔ－Ｏｒｉｅｎｔｅｄ　Ｄａｔａｂａｓｅ：ＯＯＤＢ）システムにおいても、クライアントーサーバ型のシステム構成を適用することが一般的である。この場合、データベース管理システム（Ｄａｔａｂａｓｅ　Ｍａｎａｇｅｍｅｎｔ　Ｓｙｓｔｅｍ：ＤＢＭＳ）をサーバとし、データベース管理システム（ＤＢＭＳ）で管理されるデータにアクセスするアプリケーションプログラム（Ａｐｐｌｉｃａｔｉｏｎ　Ｐｒｏｇｒａｍ：ＡＰ）をクライアントとしている。
【０００３】オブジェクト指向データベース（ＯＯＤＢ）システムで扱われるオブジェクトには、一般に「動的状態」と「永続状態」がある。オブジェクトの「動的状態」とは、通常、アプリケーションプログラム（ＡＰ）がオブジェクトに対して操作を行なえるように、計算機の主記憶上におかれた状態を指している。「動的状態」は、揮発的な性質を有している。主記憶の内容が失われるとき、例えば、アプリケーションプログラム（ＡＰ）プロセスの終了やデータベースシステムの停止などにより、「動的状態」は失われる。
【０００４】「永続状態」とは、「動的状態」を再生するために退避された状態であり、通常、ディスクなどの二次記憶装置に保持される。なお、オブジェクトデータを「永続状態」から「動的状態」へ移すことを「活性化」（ａｃｔｉｖａｔｅ）という。また、「動的状態」でアプリケーションプログラム（ＡＰ）からの操作が可能な状態を、「活性化状態」という。
【０００５】オブジェクトの「永続状態」を可能にする，即ち、オブジェクトの永続性を実現するための標準仕様として、オブジェクト・マネージメント・グループ社（Ｏｂｊｅｃｔ　Ｍａｎａｇｅｍｅｎｔ　Ｇｒｏｕｐ　Ｉｎｃ．（ＯＭＧ））によって規定されたオブジェクト永続性サービス仕様（Ｏｂｊｅｃｔ　Ｐｅｒｓｉｓｔｅｎｃｅ　Ｓｅｒｖｉｃｅ　Ｓｐｅｃｉｆｉｃａｔｉｏｎ（ＯＰＳＳ））が知られている。オブジェクト永続性サービス仕様（ＯＰＳＳ）は、オブジェクトの永続状態を管理するための機構とインタフェイスを与えるガイドラインである。例えば、オブジェクト永続性サービス仕様（ＯＯＰＳ）によれば、データストア（ｄａｔａ　ｓｔｏｒｅ）とは、オブジェクトの永続状態のための記憶を与える機構をいい、永続識別子（Ｐｅｒｓｉｓｔｅｎｔ　Ｉｄｅｎｔｉｆｉｅｒ：ＰＩＤ）とは、オブジェクトの永続状態の記憶場所を識別する情報を保持する値をいう。
【０００６】複数のオブジェクトを動的および永続的に取り扱うときは、個々のオブジェクトをシステム内で一意に識別する機構が必要になる。オブジェクト・データベース・マネージメント・グループ社（Ｏｂｊｅｃｔ　Ｄａｔａｂａｓｅ　Ｍａｎａｇｅｍｅｎｔ　Ｇｒｏｕｐ（ＯＤＭＧ））が規定するオブジェクト指向データベース（ＯＯＤＢ）の標準仕様のデータモデルによれば、“Ｔｈｅ　ＯｂｊｅｃｔＤａｔａｂａｓｅ　Ｓｔａｎｄ

ard：ODMG－93 Release 1.1”，Cattel，R. G. G. ed. Morgan KaufmannPublishers，1994に記載されているように、データベースシステム内の個々のオブジェクトを一意に識別するために、オブジェクト識別子（Object Identifier： OID）が用いられる。

【0007】オブジェクト間の「関連」（relationship）は、オブジェクト指向データモデルにおける特長的なモデリングの手段である。「OMT」と呼ばれる一般的なオブジェクト設計技法によれば、「オブジェクト指向方法論 OMTモデル化と設計」，J．Rumbaugh他、羽生田栄一監訳、トッパン（1992年）に記載されているように、「関連」とは、オブジェクトの関係を確立するための手段であり、「リンク」（link）とは、オブジェクト間の物理的あるいは概念的な結合である。「関連」は、同一の意味を持つ「リンク」の集まり（定義）である。

【0008】一般的な関連（リンク）機能の実装方法としては、オブジェクト識別子（OID）をオブジェクトデータ内部に保持する方法がある。データ内部に保持しているオブジェクト識別子（OID）に一意に対応するオブジェクトを、関連付けられたリンク先オブジェクトと解釈する。

【0009】関連は本質的に双方向性を持つ。例えば、「人」と「会社」の間に関連「勤務する」を定義する場合、「勤務する」の逆関連「雇う」は「会社」と「人」との間の関連であり、どちらの方向をたどることも等しく意味を持ち、根底にある同一の関連を示す。

【0010】また、関連には、多対多などの「多重度」の概念があり、いくつのオブジェクトがリンクで関連付けられるかが示される。例えば、「会社」が複数の「従業員」を「雇う」場合、この関連の多重度は1対多となる。

【0011】従来のオブジェクト指向データベース（OODB）システムでは、オブジェクト識別子（OID）がシステム内で一意となることを保証する。そのために、オブジェクト識別子（OID）の発行管理は、オブジェクト指向データベース（OODB）システム全体を管理するデータベース管理システム（DBMS）サーバで行なっている。

【0012】従って、クライアントアプリケーションで生成したオブジェクトにオブジェクト識別子（OID）を割り付けるには、データベース管理システム（DBMS）サーバとの通信が必要になる。

【0013】一方、オブジェクト間のリンクを管理するときには、一般にリンク先の情報として、オブジェクト識別子（OID）を用いる。そのため、新規オブジェクトにリンクを設定する際には、先にオブジェクト生成をデータベース管理システム（DBMS）サーバに伝えてオブジェクト識別子（OID）を割り付けてから、リンク設定処理を行なわなければならない。

【0014】

【発明が解決しようとする課題】この一連の処理を簡潔に行なう方式としては、まずオブジェクト生成をデータベース管理システム（DBMS）サーバと通信してオブジェクト識別子（OID）を発行し、発行したオブジェクト識別子（OID）を受け取ってクライアントでリンク設定を行ない、リンク設定に伴う更新要求について別途データベース管理システム（DBMS）サーバと通信することが考えられる。

【0015】しかし、このようにして、リンクを管理する方式では、通信回数が増加するという問題がある。通信は処理コストが高いため、上記のような処理方式では、クライアントーサーバ型システムにおいて、クライアントの要求を効率良く処理することができず、高性能で動作することができない。

【0016】オブジェクト間のリンクを効率良く扱うクライアントーサーバ型オブジェクト指向データベース（OODB）システムの例として、例えば、オブジェクトデザイン（Object Design）社の製品として、オブジェクト・ストア（Object Store）が知られている。オブジェクト・ストアでは、オブジェクトの識別については、メモリ上のポインタを用いており、オブジェクト間の関連については、リンク先をポインタで直接的に表わしている。このポインタを永続的に使用できるように、オペレーティングシステム（OS）のページングおよび仮想記憶の機構を応用した「仮想記憶マッピングアーキテクチャ」と呼ばれる方法を採用している。

【0017】この方法では、クライアントでローカルにデータページ中のオブジェクトの格納構成が決定され、サーバではそのページの状態がそのままデータストアに格納される。従って、それぞれのクライアント毎に扱うオブジェクトをデータストア中で物理的に近い位置にまとめて格納（クラスタリング）することで、個々のクライアントの処理でのアクセス効率の向上が図れる。

【0018】しかし、このような格納構成では、個々のクライアントの処理とは無関係にオブジェクト操作を行なうような場合，例えば、データベース全体を対象にして、あるデータ構造を持つ（あるタイプに属する）オブジェクトを全件検索する場合、物理的に離れた位置のデータページへのアクセスが多く、特に、検索のためのインデクスなど特殊な機構を用いていない場合は、すべてのデータページを検索対象にしなければならず、効率が悪いものである。

【0019】「仮想記憶マッピングアーキテクチャ」による方法では、複数のクライアントが個別に扱うオブジェクト群をまとめて格納制御することは困難である。

【0020】本発明の目的は、クライアントーサーバ型

で構成されるオブジェクト指向データベースシステムにおいて、システム内でのオブジェクト識別性を保証しつつ、サーバと通信を行うことなく、クライアントアプリケーションでのオブジェクト操作を効率よく処理できるオブジェクト管理方法を提供するにある。

【００２１】

【課題を解決するための手段】上記の目的を達するため、本発明は、データベース管理システムクライアントは、クライアントアプリケーションで生成したオブジェクトを一時的な仮オブジェクト識別子で管理し、上記オブジェクトについて反映要求をされたデータベース管理システムサーバは、上記仮オブジェクト識別子をデータベースシステムで一意となる正式オブジェクト識別子に変換して、上記オブジェクトのデータストアへの格納を制御するようにしたものであり、かかる方法により、クライアントアプリケーションでのオブジェクト操作を、サーバとの通信を行うことなく、処理し得るものとなる。

【００２２】上記オブジェクト管理方法において、好ましくは、上記データベース管理システムクライアントは、上記データベース管理システムサーバに対して上記オブジェクトについて反映要求をする際に、反映要求を指示する情報に、上記仮オブジェクト識別子から上記正式オブジェクト識別子に変換するための、補足的なオブジェクト識別子変換情報を付加するようにしたものであり、かかる方法により、正式オブジェクト識別子への変換を効率よく行い得るものとなる。

【００２３】上記オブジェクト管理方法において、好ましくは、上記データベース管理システムクライアントは、上記オブジェクトを生成順の番号で管理し、この生成順の番号に基づいて上記仮オブジェクト識別子をそれぞれのオブジェクトに割り付けるようにしたものであり、かかる方法により、仮オブジェクト識別子の管理が容易となる。

【００２４】上記オブジェクト管理方法において、好ましくは、上記データベース管理システムクライアントは、複数のオブジェクトを一括して上記データベース管理システムサーバに送信して、反映要求をするようにしたものであり、かかる方法により、クライアントーサーバ間の通信回数を削減できるものとなる。

【００２５】上記オブジェクト管理方法において、好ましくは、上記複数のオブジェクト間にリンクを設定し、上記反映要求を指示する情報に、リンク先情報を設定する部分更新情報を付加するようにしたものである。

【００２６】

【発明の実施の形態】以下、本発明の一実施形態によるオブジェクト管理方法について、図１乃至図１９を用いて説明する。図１は、本発明の一実施形態によるオブジェクト管理方法を適用したオブジェクト指向データベース（ＯＯＤＢ）システムの構成図である。

【００２７】クライアントアプリケーション１０１ａ，１０１ｂは、ネットワーク１０２を介して、データベース管理システム（ＤＢＭＳ）サーバ１０３と通信し、データストア１０４に格納されるオブジェクトにアクセスするクライアントーサーバ型でシステムを構成している。

【００２８】このシステムは、通信回数を増やすことなく、クライアントでのオブジェクト操作（特に、オブジェクト生成とオブジェクト間リンク設定）を行なえるように構成されている。そして、通信を回避するための基本的な処理方式として、以下の（１）～（３）を備えている。

【００２９】（１）仮オブジェクト識別子（仮ＯＩＤ）の発行

クライアントアプリケーション１０１ａ，１０１ｂは、データベース管理システム（ＤＢＭＳ）サーバ１０３と通信せずに、新規生成したオブジェクトに対して、仮のオブジェクト識別子（ＯＩＤ）を発行し、割り付ける。ここでは、この仮のオブジェクト識別子（ＯＩＤ）のことを「仮ＯＩＤ」と呼ぶことにする。

【００３０】この「仮ＯＩＤ」は、データベース管理システム（ＤＢＭＳ）サーバ１０３がシステム内で一意になるように正式に発行するオブジェクト識別子（ＯＩＤ）である「正式ＯＩＤ」と対比して用いるものである。

【００３１】仮ＯＩＤは、生成順の通番を、クライアントアプリケーション１０１ａ，１０１ｂが、オブジェクト識別子（ＯＩＤ）の内部データに取り込んで作成する。

【００３２】クライアントアプリケーション１０１ａ，１０１ｂにおいて、オブジェクト操作対象となったオブジェクトを識別する値および操作要求の内容は、仮ＯＩＤを用いて記録される。

【００３３】（２）更新反映情報の一括送信と、ＯＩＤ変換情報の付加

クライアントアプリケーション１０１ａ，１０１ｂにおけるオブジェクト操作に対応するデータベース管理システム（ＤＢＭＳ）サーバ１０３へ反映要求は、トランザクションコミットを契機に一括して送信する。

【００３４】反映要求の通信電文には、仮ＯＩＤを正式ＯＩＤに効率良く変換するための補足的な情報（ＯＩＤ変換情報）を付加しておく。

【００３５】（３）オブジェクト識別子（ＯＩＤ）を変換してデータストアへ反映

データベース管理システム（ＤＢＭＳ）サーバ１０３は、上記のＯＩＤ変換情報に従い、オブジェクトの生成通番を元に、仮ＯＩＤを効率良く正式ＯＩＤに変換し、通信電文中の仮ＯＩＤ中のオブジェクト操作要求をデータストアに反映する。

【００３６】データストアへの格納は、データベース管

理システム（ＤＢＭＳ）サーバ１０３が制御し、クラスタリングなどによりアクセス効率の向上を図る。

【００３７】次に、図１に示したクライアントアプリケーション１０１ａ，１０１ｂ及びデータベース管理システム（ＤＢＭＳ）サーバ１０３の構成について説明する。

【００３８】最初に、クライアントアプリケーション１０１ａ，１０１ｂの構成について説明する。クライアントアプリケーション１０１のアプリケーションプログラム（ＡＰ）１０５は、データベース（ＤＢ）にアクセスする手順を指示する。アプリケーションプログラム（ＡＰ）１０５は、データベース管理システム（ＤＢＭＳ）クライアント１０６が提供するアプリケーションプログラムインタフェイス（Ａｐｐｌｉｃａｔｉｏｎ Ｐｒｏｇｒａｍ Ｉｎｔｅｒｆａｃｅ：ＡＰＩ）の関数群を用いて記述される。クライアントアプリケーション１０５には、データベース管理システム（ＤＢＭＳ）クライアント１０６がバインドされる。

【００３９】データベース管理システム（ＤＢＭＳ）クライアント１０６のオブジェクト管理部１０７は、クライアントアプリケーション１０１が扱うオブジェクト１０８を管理する。オブジェクト１０８は、オブジェクト管理部１０７のクライアントオブジェクトキャッシュ１０９上に保持される。

【００４０】オブジェクト管理部１０７の中のオブジェクト状態管理テーブル１１０は、クライアントオブジェクトキャッシュ１０９上のオブジェクト１０８の状態を管理する。オブジェクト状態管理テーブル１１０のエントリは、各オブジェクト１０８に、１つずつ対応付けられる。

【００４１】データベース管理システム（ＤＢＭＳ）クライアント１０６の中のディクショナリ定義管理部１１１は、オブジェクト１０８のデータ構造（タイプ）およびアクセス方法の定義の集まりであるディクショナリ情報を管理する。アプリケーションプログラム１０５の動作に必要なディクショナリ定義情報が、クライアントアプリケーション１０１にバインドされるものとする。

【００４２】データベース管理システム（ＤＢＭＳ）クライアント１０６の中の仮ＯＩＤ管理部１１２は、クライアントアプリケーション１０１で生成する個々のオブジェクト１０８を識別する仮ＯＩＤの発行を管理する。仮ＯＩＤは、仮ＯＩＤ管理部１１２で管理される生成順による生成通番１１３をもとに作成される。

【００４３】データベース管理システム（ＤＢＭＳ）クライアント１０６の通信管理部１１４では、データベース管理システム（ＤＢＭＳ）サーバ１０３との通信を管理し、クライアントアプリケーション１０１におけるオブジェクト操作をデータベース（ＤＢ）に反映させるための更新電文１１５を管理する。

【００４４】次に、データベース管理システム（ＤＢＭＳ）サーバ１０３の構成について説明する。データベース管理システム（ＤＢＭＳ）サーバ１０３は、トランザクション管理やオブジェクト管理など、データベースシステムが提供する各機能を実現するためのプロセスの集まりから構成されている。

【００４５】データベース管理システム（ＤＢＭＳ）サーバ１０３の中のシステム制御部１１６は、データベース（ＤＢ）システムを構成するプロセス群を監視し、システム全体の制御を行なう。

【００４６】システム制御部１１６の中のトランザクション管理部１１７は、各クライアントアプリケーション１０１で実行されるトランザクションの管理を行なう。

【００４７】サーバアプリケーション１１８ａ，１１８ｂは、クライアントアプリケーション１０１ａ，１０１ｂにそれぞれ１つずつ対応するプロセスで、クライアントアプリケーション１０１からの要求を受け付け、データベース管理システム（ＤＢＭＳ）サーバ１０３の各モジュール（管理部，制御部）に要求を伝える。

【００４８】データベース管理システム（ＤＢＭＳ）サーバ１０３の中のオブジェクト管理部１１９は、サーバアプリケーション１１８のオブジェクト操作要求を受け付けて処理する。サーバオブジェクト管理部１１９は、クライアントアプリケーション１０１のオブジェクト１０８に対応するように、データベース管理システム（ＤＢＭＳ）サーバ１０３のオブジェクトキャッシュ１２０上でオブジェクト１２１を操作し、このオブジェクト１２１の状態は、オブジェクト管理情報テーブル１２２により管理する。

【００４９】データベース管理システム（ＤＢＭＳ）１０３の中のサーバＯＩＤ管理部１２３は、データベース（ＤＢ）システムにおける正式ＯＩＤを発行し、一意性を保証するように管理する。

【００５０】データベース管理システム（ＤＢＭＳ）１０３の中のサーバＯＩＤ変換管理部１２４は、クライアントアプリケーション１０１が発行した仮ＯＩＤと、正式ＯＩＤとの対応関係を管理する。対応関係は、サーバＯＩＤ変換管理部１２４の中のＯＩＤ変換テーブル１２５を用いて管理する。ＯＩＤ変換テーブル１２５の詳細については、図１９を用いて後述する。

【００５１】格納制御部１２６は、オブジェクトデータを格納するデータストア１０４とのデータ入出力を管理する。

【００５２】データストア１０４は、オブジェクト１２１の永続状態のための記憶を与える機構である。

【００５３】通信管理部１０７は、クライアントアプリケーション１０１からメッセージを受け取るバッファの管理を行う。

【００５４】次に、図２を用いて、本実施形態によるオブジェクト管理方法の処理方式について説明する。図２は、本発明の一実施形態によるオブジェクト管理方法に

おけるクライアントアプリケーションでのアプリケーションプログラムの処理フローの概念を示すフローチャートである。なお、詳細な処理フローについては、図３以下を用いて後述する。

【００５５】本実施形態においては、例えば、２つのオブジェクトを作成し、それらのオブジェクト間にリンクを設定するオブジェクト操作のプログラムの処理について説明する。

【００５６】ステップ２０１において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、データベース（ＤＢ）接続するためのデータベース管理システム（ＤＢＭＳ）クライアント１０６のアプリケーションプログラムインタフェイス（ＡＰＩ）関数をコールし、クライアントアプリケーション１０１とデータベース管理システム（ＤＢＭＳ）サーバ１０３との通信を確立する。ステップ２０１の詳細については、図３を用いて後述する。

【００５７】ステップ２０２において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、トランザクションを開始するＡＰＩ関数をコールする。ステップ２０２の詳細については、図６を用いて後述する。この後、トランザクション開始から、次のトランザクションコミットまで間の処理について、データベース（ＤＢ）システムで整合性が保証される。

【００５８】ステップ２０３において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト１０８を生成するＡＰＩ関数をコールし、「オブジェクト１」を生成する。ステップ２０３の詳細については、図７，８を用いて後述する。

【００５９】ここで、オブジェクト生成ＡＰＩの入力として、生成するオブジェクトのタイプを指定する手段があるものとする。例えば、ディクショナリ定義管理部１１１に定義されたタイプ識別子を指定するものとする。タイプ識別子は、データベース（ＤＢ）システムの一般的なディクショナリ定義操作により与えられ、指定されたタイプ識別子に対応するタイプ定義に基づくデータ構造のオブジェクトが生成されるものとする。

【００６０】ステップ２０４において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ステップ２０３と同様にして、「オブジェクト２」を生成する。ここで、「オブジェクト１」と「オブジェクト２」で指定するタイプは、同じであっても、異なっていてもよい。ただし、それぞれのタイプ間に１対１の双方向関連が定義されているものとする。

【００６１】ステップ２０５において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト間リンク設定ＡＰＩ関数をコールし、「オブジェクト１」と「オブジェクト２」の間にリンクを設定する。ステップ２０５の詳細については、図９，１０，１１を用いて後述する。ここで、リンク設定ＡＰＩの入力として、ディクショナリに定義された関連の関連識別子を指定するものとする。この指定により、関連の定義情報に従い、オブジェクト間のリンクが設定されるものとする。

【００６２】ステップ２０６において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、トランザクションコミットするＡＰＩ関数をコールする。ステップ２０６の詳細については、図１２を用いて後述する。これにより、ステップ２０３以降の処理を、データベース（ＤＢ）に反映させる。

【００６３】ステップ２０７において、データベース（ＤＢ）接続を解除するＡＰＩ関数をコールし、クライアントアプリケーション１０１とデータベース管理システム（ＤＢＭＳ）サーバ１０３との通信を解除する。ステップ２０７を終えた後、アプリケーションプログラムを終了する。

【００６４】図３を用いて、図２のステップ２０１に示したデータベース管理システム（ＤＢＭＳ）クライアントにおけるデータベース（ＤＢ）接続ＡＰＩの処理の詳細について説明する。図３は、本発明の一実施形態によるオブジェクト管理方法におけるデータベース（ＤＢ）接続ＡＰＩの処理手順を示すフローチャートである。

【００６５】ステップ３０１において、データベース管理システム（ＤＢＭＳ）クライアント１０６のオブジェクト管理部１０７は、制御情報のデータ領域を確保し、初期化する。

【００６６】ここで、データベース管理システム（ＤＢＭＳ）クライアント１０６の制御データ構造について、図４を用いて説明する。図４は、本発明の一実施形態によるオブジェクト管理方法において用いるデータベース管理システム（ＤＢＭＳ）クライアントで取り扱う制御データの構造の説明図である。

【００６７】この制御情報の構成により、クライアントアプリケーション１０１で一意となる仮ＯＩＤの発行を管理し、トランザクションコミット時に、一括して更新反映要求を行なうための情報を保持する。

【００６８】データベース管理システム（ＤＢＭＳ）クライアント１０６の機能のための制御情報として、クライアント管理制御ヘッダ４０１，オブジェクト管理制御ヘッダ４０２，仮ＯＩＤ管理制御ヘッダ４０３，通信管理制御ヘッダ４０４がある。これらの制御情報のアドレスは、クライアント管理制御ヘッダアドレスとして、データベース管理システム（ＤＢＭＳ）クライアントの各モジュールに対してグローバルな値に保持され、各モジュールがこれらの制御情報にアクセスする。

【００６９】クライアント管理制御ヘッダ４０１は、クライアントアプリケーション１０１のプロセス全体を管理するために必要な情報（ステータス）を保持している。例えば、アプリケーションプロセスの状態（ＤＢ接続状態，トランザクション実行状態）を保持している。

【００７０】オブジェクト管理制御ヘッダ４０２は、オブジェクト管理部１０７を制御する情報を保持している。オブジェクト管理制御ヘッダ４０２は、キャッシュ制御ヘッダ４０５のアドレス、およびオブジェクト状態管理制御ヘッダ４０６のアドレスを保持している。

【００７１】キャッシュ制御ヘッダ４０５は、オブジェクトキャッシュ１０９を管理するために使用し、オブジェクト１０８のデータのキャッシングの管理に必要な情報を保持している。キャッシュ制御ヘッダ４０５は、例えば、オブジェクトキャッシュ１０９のキャッシュ領域の先頭アドレス，利用可能領域の先頭アドレスを保持している。

【００７２】オブジェクト状態管理テーブル制御ヘッダ４０６は、オブジェクト状態管理テーブル１１０を管理する。オブジェクト状態管理テーブル１１０のエントリを割り付けるために、オブジェクト状態管理テーブル１１０のテーブル領域の先頭アドレス，利用可能テーブルエントリの先頭アドレスなどの情報を保持している。

【００７３】仮ＯＩＤ管理制御ヘッダ４０３は、仮ＯＩＤを発行するための情報を管理する。仮ＯＩＤの元となる生成順通番カウンタ１１３を保持している。

【００７４】通信管理制御ヘッダ４０４は、通信管理情報を制御するために用い、通信電文を保持する領域である通信バッファ４０８の先頭アドレスなどを保持する。

【００７５】本実施形態のシステムでは、クライアントアプリケーション１０１で行なったオブジェクト操作のデータベース管理システム（ＤＢＭＳ）サーバ１０３に対する反映は、コミット時に一括して行なう。反映する要求の内容を、データベース管理システム（ＤＢＭＳ）クライアント１０６で一時的に更新情報として保持し、オブジェクト状態管理エントリ４０７からポイントしておく。反映が必要な更新要求情報を効率良くに得るために、反映が必要なオブジェクト状態管理エントリ４０７を、オブジェクト状態管理制御ヘッダ４０２からポイントし、さらに、複数の更新対象のオブジェクト状態管理エントリ４０７を得られるように各エントリをポインタで連結しておく。ここでは、この連結のことを「反映要求チェイン４０９」と称する。

【００７６】次に、ステップ３０２において、オブジェクト管理部１０７は、仮ＯＩＤ管理部１１２を初期化し、仮ＯＩＤ管理部１１２は、生成通番１１３を初期化する。

【００７７】ステップ３０３において、オブジェクト管理部１０７は、通信管理部１１４を初期化し、以降のデータベース管理システム（ＤＢＭＳ）サーバ１０３への通信要求に備える。

【００７８】ステップ３０４において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、データベース（ＤＢ）接続要求電文を作成する。

【００７９】ここで、本実施形態において用いる通信電文のデータ構造について、図５を用いて説明する。図５は、本発明の一実施形態によるオブジェクト管理方法においてクライアントアプリケーションが作成する通信電文のデータ構造の説明図である。

【００８０】通信電文５０１は、電文制御ヘッダ５０２と、複数（ｎ個）のコマンドパラメタブロック５０３から構成されている。電文制御ヘッダ５０２は、通信電文５０１を制御するために必要な以下の情報を保持している。

【００８１】（１）「電文の形式」（サーバへの送信データであることを示す値など）
（２）「電文のサイズ」
（３）「先頭ブロックオフセット」（先頭のコマンドパラメタブロック５０３の位置を示すオフセット値；通信電文５０１の先頭からのオフセット値として与えられている）
（４）「ブロック総数（ｎ）」（電文５０１に保持するコマンドパラメタブロック５０３の総数）
コマンドパラメタブロック５０３は、ブロック制御ヘッダ５０４と、ブロックデータ５０４から構成されている。ブロック制御ヘッダ５０４は、各コマンドパラメタブロック５０３を制御する情報を有している。ブロック制御ヘッダ５０４は、例えば、「ブロック形式」，「ブロックサイズ」（ブロックのデータ長），「コマンド識別子」（データベース（ＤＢ）接続要求であることなどを示す値など），「ブロックデータオフセット」（ブロックデータの位置を示すオフセット値；コマンドパラメタブロック５０３の先頭からのオフセット値），「次ブロックのオフセット」（次のブロックの位置を示すオフセット値；通信電文５０１の先頭からのオフセット値）を保持している。

【００８２】以上説明した構造を用いることにより、例えば、データベース（ＤＢ）接続要求の電文メッセージでは、電文制御ヘッダ５０２内の「ブロック総数」を「１」とし、最初のブロック１のブロック制御ヘッダ５０４内の「コマンド識別子」に「ＤＢ接続」を示す値を設定するようにしている。

【００８３】ステップ３０５において、通信管理部１１４は、データベース管理システム（ＤＢＭＳ）サーバ１０３へ通信電文５０１を送信する。ステップ３０５を終えた後、データベース（ＤＢ）接続処理を終了する。

【００８４】ここで、プロセス間通信の手段および形態については、本実施形態の方法を実装するために特別な機能を必要とせず、周知の機構を用いてよい。

【００８５】図６を用いて、図２のステップ２０２に示したデータベース管理システム（ＤＢＭＳ）クライアントにおけるトランザクション開始ＡＰＩの処理の詳細について説明する。図６は、本発明の一実施形態によるオブジェクト管理方法におけるトランザクション開始ＡＰＩ関数コールの処理手順を示すフローチャートである。

【００８６】ステップ６０１において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト管理部１０７にリセット要求し、オブジェクト管理部１０７がリセットする（制御情報を初期状態に戻し）ことにより、後のオブジェクト操作要求に備える。

【００８７】ステップ６０２において、オブジェクト管理部１０７は、仮ＯＩＤ管理部１１２にリセット要求し、仮ＯＩＤ管理部１１２がリセットすることにより、後のオブジェクト生成時の仮ＯＩＤ発行に備える。

【００８８】ステップ６０３において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、トランザクション開始要求の通信電文を作成する。通信電文は、通信管理部１１４の通信バッファ領域に作成される。

【００８９】ステップ６０４において、通信管理部１１４は、作成した通信電文を、データベース管理システム（ＤＢＭＳ）サーバ１０３へ送信する。データベース管理システム（ＤＢＭＳ）サーバ１０３は、トランザクション管理部１１７に、トランザクション開始要求が伝えられ、クライアントアプリケーション１０１のプロセスでのトランザクションが認識され、管理対象となる。

【００９０】ステップ６０４を終えた後、この処理を終了する。

【００９１】次に、図７及び図８を用いて、図２のステップ２０３に示したデータベース管理システム（ＤＢＭＳ）クライアントにおけるオブジェクト生成ＡＰＩの処理の詳細について説明する。図７は、本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト生成ＡＰＩの処理手順を示すフローチャートであり、図８は、生成されるオブジェクトのデータ構造の説明図である。

【００９２】図７に示すステップ７０１において、オブジェクト管理部１０７は、ディクショナリに定義されているタイプ定義情報を参照する。参照するタイプ定義対象は、アプリケーションプログラム１０５からオブジェクト生成ＡＰＩのパラメタで指定されており、このタイプ識別子に対応するタイプ定義情報を、ディクショナリ定義管理部１１１から取得する。

【００９３】ステップ７０２において、オブジェクト管理部１０７は、オブジェクトキャッシュ１０９上にデータ領域を取得する。データ領域の大きさは、ステップ７０１で取得したタイプ定義情報を参照して、必要なサイズを算出する。

【００９４】ステップ７０３において、オブジェクト管理部１０７は、ステップ７０１で取得したタイプ定義情報に従って、オブジェクト１０８を初期化する。オブジェクト１０８のデータ構造には、リンク先情報を保持する構造があり、このステップにおいて、リンク先情報を初期化する。

【００９５】ステップ７０４において、オブジェクト管理部１０７は、このオブジェクトの状態を管理するために、オブジェクト状態管理テーブル１１０のエントリを割り付ける。

【００９６】ステップ７０５において、オブジェクト管理部１０７は、ステップ７０４で取得したオブジェクト状態管理テーブル１１０のエントリを初期化する。ここで、オブジェクト状態管理テーブル１１０のエントリの中のオブジェクト状態を保持する領域のフラグを、新規状態であることを示すＯＮに設定する。

【００９７】ステップ７０６において、オブジェクト管理部１０７は、このオブジェクト状態管理テーブル１１０のエントリを、反映要求チェイン４０９につなぐ。反映要求チェイン４０９は、次のエントリをポインタで指示する。

【００９８】ステップ７０６を終えた後、オブジェクト生成ＡＰＩ処理を終了する。

【００９９】次に、図８を用いて。データベース管理システム（ＤＢＭＳ）クライアント１０６におけるオブジェクト１０８のデータ構造について説明する。オブジェクト１０８は、オブジェクト制御ヘッダ８０１と、オブジェクトデータ領域８０２から構成されている。オブジェクト制御ヘッダ８０１は、オブジェクトデータを制御情報するために、以下の情報から構成されている。

【０１００】（１）「オブジェクト状態管理テーブル１１０のオブジェクト状態管理エントリ８０６のアドレス」

（２）「オブジェクトの種別」

（３）「オブジェクト全体のデータサイズ」

（４）「自分自身のＯＩＤ」（このオブジェクトに割り付けられたＯＩＤ）

クライアントアプリケーション１０１で生成された時点では、「自分自身のオブジェクト識別子（ＯＩＤ）」は設定されていない。仮ＯＩＤは、当該オブジェクト操作において、オブジェクト識別子（ＯＩＤ）が必要になるとき、例えば、リンク設定を行なうときに発行され、割り付けられる。リンク設定時の仮ＯＩＤの割り付けについては、図９を用いて後述する。

【０１０１】「自分自身のＯＩＤ」の中は、「オブジェクト識別子（ＯＩＤ）の形式」（「仮」であるか「正式」であるかを示す値）と、一意性のための「識別値」から構成されている。

【０１０２】仮ＯＩＤ８０３の場合には、「ＯＩＤ形式」には、仮オブジェクト識別子（ＯＩＤ）形式であることを示す値が設定され、「識別値」には、割付対象オブジェクトの生成時の生成通番が設定される。

【０１０３】正式ＯＩＤ８０４の場合には、「ＯＩＤ形式」には、正式オブジェクト識別子（ＯＩＤ）形式であることを示す値が設定され、「識別値」には、データベース管理システム（ＤＢＭＳ）サーバ１０３のＯＩＤ変換管理部１２４において、データベース（ＤＢ）システ

ム内で一意になるように作成された値が設定される。ここでは、オブジェクトデータ格納位置情報に基づく永続識別子ＰＩＤを識別値として用いる。

【０１０４】オブジェクトデータ領域８０２には、オブジェクトの内容を示す「オブジェクトデータ」を保持している。この領域に、リンク先情報８０５も保持される。リンク先情報８０５には、「リンク先ＯＩＤ」（リンク先オブジェクトのオブジェクト識別子（ＯＩＤ））が保持される。リンク先オブジェクトが生成されてトランザクションコミットする前ならば、仮ＯＩＤ８０３を保持することになる。リンク先オブジェクトがデータストアに格納されている既存のオブジェクトならば、正式ＯＩＤ８０４を保持することになる。

【０１０５】オブジェクト状態管理テーブル１１０のオブジェクト状態管理エントリ８０６は、オブジェクト１０８の状態を管理するために、以下の情報を保持している。

【０１０６】（１）「管理対象のオブジェクト識別子（ＯＩＤ）」

（２）「オブジェクト状態フラグ」（生成／更新の状態であることを示す値を保持する。新規の場合には、フラグをＯＮする）

（３）「オブジェクトデータのアドレス」（管理対象オブジェクトのオブジェクトデータ領域の先頭アドレス；オブジェクト制御ヘッダ８０１の種別の欄を先頭アドレスとしている）

（４）「部分更新情報のアドレス」（管理対象オブジェクトに対して要求された部分更新情報のアドレス）

（５）「オブジェクト識別子（ＯＩＤ）変換情報のアドレス」（管理対象オブジェクトに対して要求されたＯＩＤ変換情報のアドレス）

（６）「オブジェクト識別子（ＯＩＤ）変換テーブル登録要請フラグ」

（７）「更新チェイン４０９の続き要素エントリのアドレス」

次に、図９，１０，１１を用いて、図２のステップ２０５に示したデータベース管理システム（ＤＢＭＳ）クライアントにおけるオブジェクト間リンク設定ＡＰＩの処理の詳細について説明する。図９は、本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト間リンク設定ＡＰＩの処理手順を示すフローチャートである。

【０１０７】ステップ９０１において、オブジェクト管理部１０７は、ディクショナリ定義管理部１１１に定義されているオブジェクト間関連定義情報を参照する。対象となる関連定義は、アプリケーションプログラム１０５からリンク設定ＡＰＩのパラメタとして関連識別子で指定され、この関連識別子に対応する関連定義情報を、ディクショナリ定義管理部１１１から取得する。

【０１０８】ステップ９０２において、オブジェクト管理部１０７は、リンク元オブジェクトが新規で、かつ仮ＯＩＤが割り当てられていないかを判定する。新規であるか否かは、図８に示したオブジェクト状態管理エントリ８０６内の「状態フラグ」で判定する。新規の場合には、「状態フラグ」がＯＮになっている。仮ＯＩＤが割り当てられているか否かは、図８に示したオブジェクト制御ヘッダ８０１内の「自分自身のＯＩＤ」で、「ＯＩＤ形式」を参照することにより判定する。「ＯＩＤ形式」には、仮ＯＩＤ形式であることを示す値若しくは、正式ＯＩＤ形式であることを示す値が設定されているので、これらの値が設定されていない場合には、仮ＯＩＤが割り当てられていないと判定する。

【０１０９】仮ＯＩＤが割り当てられていない場合は、ステップ９０３に進み、仮ＯＩＤが割り当てられている場合は、ステップ９０６に進む。

【０１１０】ステップ９０３において、仮ＯＩＤ管理部１１２は、新規の仮ＯＩＤを発行する。このとき、仮ＯＩＤ管理部１１２は、生成通番１１３をインクリメントする。

【０１１１】次に、ステップ９０４において、オブジェクト管理部１０７は、リンク元オブジェクトのオブジェクト制御ヘッダ８０１およびオブジェクト状態管理エントリ８０６内に、発行した仮ＯＩＤを設定する。即ち、オブジェクト制御ヘッダ８０１の自分自身のＯＩＤに、仮ＯＩＤ８０３を設定する。生成通番１１３には、ステップ９０３でインクリメントした生成通番を設定する。また、オブジェクト状態管理エントリ８０６の「管理対象ＯＩＤ」に、仮ＯＩＤを設定する。

【０１１２】次に、ステップ９０５において、オブジェクト管理部１０７は、オブジェクト状態管理エントリ８０６内の「ＯＩＤ変換テーブル登録要請フラグ」をＯＮにする。このフラグの値に従って、データベース管理システム（ＤＢＭＳ）サーバ１０３で反映処理を行なうときに、ＯＩＤ変換テーブル１２５に、このオブジェクトについてのエントリを登録することにより、ＯＩＤ変換テーブル１２５へのエントリ登録を効率良く行なうことができる。ステップ９０５を終えた後、ステップ９０６に進む。

【０１１３】ステップ９０６からステップ９０９においては、リンク先オブジェクトについて、ステップ９０２からステップ９０５と同様して、オブジェクト識別子（ＯＩＤ）を判定し、仮ＯＩＤを割り付ける処理を行なう。

【０１１４】即ち、ステップ９０６において、オブジェクト管理部１０７は、リンク先オブジェクトが新規で、かつ仮ＯＩＤが割り当てられていないかを判定する。仮ＯＩＤが割り当てられていない場合は、ステップ９０７に進み、仮ＯＩＤが割り当てられている場合は、ステップ９１０に進む。

【０１１５】ステップ９０７において、仮ＯＩＤ管理部

１１２は、新規に仮ＯＩＤを発行する。このとき、仮ＯＩＤ管理部１１２は、生成通番１１３をインクリメントする。

【０１１６】次に、ステップ９０８において、リンク先オブジェクトのオブジェクト制御ヘッダ８０１およびオブジェクト状態管理エントリ８０６内に、発行した仮ＯＩＤを設定する。

【０１１７】次に、ステップ９０９において、オブジェクト管理部１０７は、オブジェクト状態管理エントリ８０６内の「ＯＩＤ変換テーブル登録要請フラグ」をＯＮにする。ステップ９０９を終えた後、ステップ９１０へ進む。

【０１１８】ステップ９１０において、オブジェクト管理部１０７は、当該処理でリンク元オブジェクトに対して行なう更新を記録しておくための部分更新要求情報を作成する。この更新要求情報作成処理の詳細については、図１０，１１を用いて後述する。

【０１１９】さらに、ステップ９１１において、オブジェクト管理部１０７は、ステップ９１０と同様に、当該処理でリンク先オブジェクトに対して行なう更新を記録しておくための更新要求情報を作成する。この更新要求情報作成処理の詳細についても、図１０，１１を用いて後述する。

【０１２０】ステップ９１１を終えた後、リンク設定ＡＰＩ処理を終了する。

【０１２１】次に、図１０，１１を用いて、図９のステップ９１０，９１１に示したデータベース管理システム（ＤＢＭＳ）クライアントにおける更新情報作成の処理の詳細について説明する。図１０は、本発明の一実施形態によるオブジェクト管理方法におけるリンク設定時の更新情報作成の処理手順を示すフローチャートであり、図１１は、本発明の一実施形態によるオブジェクト管理方法におけるリンク設定時の更新情報作成処理で扱われるデータ構造の説明図である。

【０１２２】る。

【０１２３】ステップ１００１において、オブジェクト管理部１０７は、ディクショナリ定義管理無１１１から取得した関連定義情報を参照する。取得した関連定義情報から、オブジェクト内にリンク先情報を保持する位置のオフセットを取得し、リンク先情報をリンク元オブジェクト内に埋め込む。ここで、リンク先情報とは、リンク先オブジェクトのＯＩＤである。

【０１２４】即ち、図１１において、オブジェクト１１０１ａをリンク先オブジェクトとし、オブジェクト１１０１ｂをリンク元オブジェクトとすると、リンク元オブジェクト１１０ｂのリンク先情報１１０２に、リンク先オブジェクト１１０１ａの「仮ＯＩＤａ」を埋め込む。

【０１２５】ステップ１００２において、オブジェクト管理部１０７は、リンク元オブジェクトから、「オブジェクト制御ヘッダ」に保持しているアドレスを参照して、オブジェクト状態管理エントリを取得する。

【０１２６】即ち、図１１において、リンク元オブジェクト１１０１ｂの「オブジェクト制御ヘッダ」の先頭に保持しているアドレスを参照して、オブジェクト状態管理テーブル１１０のオブジェクト状態管理エントリ１１０４ｂを取得する。

【０１２７】ステップ１００３において、オブジェクト管理部１０７は、リンク先オブジェクトのＯＩＤが仮ＯＩＤであるかを判定する。この判定は、ＯＩＤ内の「ＯＩＤ形式」を参照して行われる。仮ＯＩＤの場合は、ステップ１００４に進み、仮ＯＩＤでない場合は、ステップ１００６に進む。

【０１２８】即ち、図１１において、リンク先オブジェクト１１０１ａの「オブジェクト制御ヘッダ」内の「自分自身のＯＩＤ」の「ＯＩＤ形式」（図８参照）を参照して判定する。ここでは、リンク先オブジェクト１１０１ａ及びリンク元オブジェクト１１０１ｂのいづれも仮ＯＩＤであるので、ステップ１００４に進むものとする。

【０１２９】ステップ１００４において、オブジェクト管理部１０７は、リンク先オブジェクトのＯＩＤを埋め込む位置（リンク元オブジェクト内のリンク先情報）のオフセットをもとに、ＯＩＤ変換情報を作成する。このＯＩＤ変換情報に設定するオフセットは、データベース管理システム（ＤＢＭＳ）サーバ１０３が、仮ＯＩＤが埋め込まれた位置を効率良く得るために用いられる。

【０１３０】即ち、図１１において、ＯＩＤ変換情報１１０５には、更新位置オフセットが設定される。この更新位置オフセットは、リンク元オブジェクト１１０１ｂのリンク先情報１１１０２のオフセットである。

【０１３１】次に、ステップ１００５において、オブジェクト管理部１０７は、リンク元オブジェクトのオブジェクト管理管理エントリに、ＯＩＤ変換情報のアドレスを設定する。１つのオブジェクトに対して複数の仮ＯＩＤが埋め込まれる場合に対処するため、ＯＩＤ変換情報をアドレスポインタで連結し、オブジェクト状態管理エントリには、ＯＩＤ変換情報のチェインの先頭の要素（ＯＩＤ変換情報１１０５）のアドレスを保持する。

【０１３２】即ち、図１１において、リンク元オブジェクト１１０１ｂのオブジェクト管理管理エントリ１１０４ｂに、ＯＩＤ変換情報１１０５のアドレスを設定する。アドレスの設定位置は、図８において説明したように、ＯＩＤ変換要請フラグの前の位置である。複数の仮ＯＩＤが埋め込まれる場合には、図１１に示すように、次のＯＩＤ変換情報を、前のＯＩＤ変換情報１１０５のアドレスポインタで連結するようにする。

【０１３３】ステップ１００６において、オブジェクト管理部１０７は、リンク先情報を設定するための部分更新情報を作成する。部分更新情報は、「更新位置オフセット」，「更新範囲」および「更新データ内容の先頭ア

ドレス」から構成されている。１つのオブジェクトに対する複数の部分更新に対処するため、部分更新情報は、「アドレスポインタ」で連結する。オブジェクト状態管理エントリには、部分更新情報チェインの先頭要素のアドレスを保持する。

【０１３４】即ち、図１１において、部分更新情報１１０３は、「更新位置オフセット」，「更新範囲サイズ」，「更新データ内容の先頭アドレス」及び「次の部分更新情報のアドレスポインタ」から構成されている。「更新データ内容の先頭アドレス」は、リンク先情報１１０２の先頭アドレスである。オブジェクト状態管理エントリ１１０４ｂは、部分更新情報のチェインの先頭の要素（部分更新情報１１０３）のアドレスを保持している。その位置は、図８において説明したように、ＯＩＤ変換情報１１０５のアドレスの前の位置である。

【０１３５】ステップ１００７において、オブジェクト管理部１０７は、リンク元オブジェクトのオブジェクト状態管理エントリに部分更新情報のアドレスを設定する。既に、部分更新情報が作成されている場合には、新たに作成した部分更新情報を、既作成の部分更新情報のチェインに連結する。

【０１３６】即ち、図１１において、オブジェクト状態管理エントリ１１０４ｂの「ＯＩＤ変換要請フラグ」の２つ前の位置に、部分更新情報１１０３のアドレスを設定する。

【０１３７】ステップ１００８において、オブジェクト管理部１０７は、リンク元オブジェクト状態管理エントリを設定する。即ち、リンク元オブジェクト状態管理エントリの「状態フラグ」の「更新」を示す値をＯＮにする。このフラグは、後のトランザクションコミットで反映情報電文を作成するときに、当該オブジェクトに部分更新の反映が要求されていること判定するために用いる。

【０１３８】即ち、図１１において、リンク元オブジェクト１１０１ｂのオブジェクト状態管理エントリ１１０４ｂのオブジェクト状態フラグの「更新」を示す値をＯＮにする。

【０１３９】ステップ１００９において、オブジェクト管理部１０７は、リンク元オブジェクト状態管理エントリを、反映要求チェインに連結する。

【０１４０】即ち、図１１において、リンク元オブジェクト１１０１ｂのオブジェクト状態管理エントリ１１０４ｂを、反映要求チェイン４０９に連結する。

【０１４１】ステップ１００９を終えた後、更新情報作成処理を終了する。

【０１４２】ここで、図１１において、図１０において説明した更新情報作成処理を実行した結果、リンク元オブジェクト１１０１ｂのオブジェクトデータ領域内にあるリンク先情報１１０２に、リンク先オブジェクト１１０１ａに割り付けられたＯＩＤ（仮ＯＩＤａ）が設定される。この設定要求を記録しておくために、部分更新情報１１０３が割り付けられる。部分更新情報１１０３は、オブジェクト状態管理エントリ１１０４ｂの部分更新情報チェインにつながれる。

【０１４３】また、リンク先オブジェクト１１０１ａのＯＩＤが仮ＯＩＤの場合は、ＯＩＤ変換情報１１０５が作成され、オブジェクト状態管理エントリ１１０４ｂのＯＩＤ変換情報チェインにつながれる。リンク先が正式ＯＩＤを持つ場合は、このリンク設定に関しては、ＯＩＤ変換情報は不要となる。

【０１４４】なお、上述した例では、双方向関連としているため、リンク先オブジェクトについても、同様の構造をとっている。

【０１４５】次に、図１２，図１３を用いて、図２のステップ２０６に示したデータベース管理システム（ＤＢＭＳ）クライアントにおけるトランザクションコミットするＡＰＩの処理の詳細について説明する。図１２は、本発明の一実施形態によるオブジェクト管理方法におけるトランザクションコミットＡＰＩの処理手順を示すフローチャートであり、図１３は、本発明の一実施形態によるオブジェクト管理方法におけるトランザクションコミット要求の通信電文データ構造の説明図である。

【０１４６】ステップ１２０１において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト制御ヘッダからオブジェクト状態管理エントリの反映要求チェインの先頭要素アドレスを取得する。

【０１４７】即ち、図１１において、オブジェクト１１０１ｂのオブジェクト制御ヘッダからオブジェクト状態管理エントリ１１０４ｂを取得し、オブジェクト状態管理エントリ１１０４ｂから反映要求チェイン４０９の先頭アドレスを取得する。

【０１４８】ステップ１２０２において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ステップ１２０１で取得した反映要求チェインに要素があるか、すなわち、反映要求がなされているかを判定する。反映要求がなされていない場合は、ステップ１２０３に進み、反映要求がなされている場合は、ステップ１２０５に進む。

【０１４９】ここで、リンク先オブジェクト１１０１ａとリンク元オブジェクト１１０１ｂがリンクされているので、反映要求がなされているとして、ステップ１２０５に進んで説明する。

【０１５０】ステップ１２０５において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、仮ＯＩＤ管理部の生成通番を、ＯＩＤ変換総数として、通信電文に設定する。この値は、サーバでＯＩＤ変換が必要な総数を伝え、ＯＩＤ変換テーブルの領域を初期化するために用いられる。

【０１５１】ここでは、オブジェクト１１０１ａ，１１０１ｂの２つのオブジェクトに対して仮ＯＩＤが発行さ

れており、この2つのオブジェクトを一括送信する場合について考えると、図１３に示したコミット要求の通信電文１３０１の変換ＯＩＤ総数３０２には、”２”が設定される。

【０１５２】ステップ１２０６において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、反映要求チェイン４０９をたどり、反映が必要なオブジェクト状態管理エントリ１１０４ｂのアドレスを取得する。

【０１５３】ステップ１２０７において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ステップ１２０６で取得したオブジェクト状態管理エントリ内の「状態フラグ」で「新規」を示す値がＯＮであるかを判定する。「新規フラグ」がＯＮの場合には、ステップ１２０８に進み、オブジェクト生成要求電文を作成する。オブジェクト生成要求電文は、図１３のコマンドパラメタ１３０３に示されるように作成されるが、このオブジェクト生成要求電文作成の詳細については、図１４を用いて後述する。「新規フラグ」がＯＮでない場合には、ステップ１２０９に進む。

【０１５４】ステップ１２０９において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、反映要求チェインの続きがあるかを判定する。続きがある場合には、ステップ１２０６に戻り、ステップ１２０６からステップ１２０９までの処理を繰り返すことにより、すべての生成要求についての電文が作成される。２回目のステップ１２０８において、図１３のコマンドパラメタ１３０４に示されるオブジェクト生成要求電文が作成される。ステップ１２０９で続きがない場合は、ステップ１２１０に進む。

【０１５５】ステップ１２１０において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、反映要求チェインの先頭に戻る。即ち、図１１に示したオブジェクト制御ヘッダ１１０１ｂを参照して、オブジェクト状態管理テーブル１１０のオブジェクト状態管理エントリ１１０４ｂのアドレスを取得する。

【０１５６】ステップ１２１１において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、反映要求チェインをたどり、反映が必要なオブジェクト状態管理エントリのアドレスを取得する。即ち、図１１に示した反映要求チェイン４０９をたどり、次のオブジェクト状態管理エントリのアドレスを取得する。

【０１５７】ステップ１２１２において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト状態管理エントリ内の状態フラグで更新を示す値がＯＮであるかを判定する。更新フラグがＯＮの場合、ステップ１２１３に進み、オブジェクト更新要求電文を作成する。オブジェクト更新要求電文は、図１３のコマンドパラメタ１３０５に示されるように作成されるが、このオブジェクト更新要求電文作成の詳細については、図１５を用いて後述する。更新フラグがＯＮでない場合、ステップ１２１４に進む。

【０１５８】ステップ１２１４において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、反映要求チェインの続きがあるかを判定する。続きがある場合は、ステップ１２１１に戻り、ステップ１２１１からステップ１２１４までの処理を繰り返すことにより、すべての更新要求についての電文が作成される。２回目のステップ１２１３において、図１３のコマンドパラメタ１３０８に示されるオブジェクト更新要求電文が作成される。ステップ１２１４で続きがない場合は、ステップ１２１５に進む。これにより、すべての更新要求についての電文が作成される。

【０１５９】ステップ１２１５において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、トランザクションコミット要求の電文を作成する。データベース管理システム（ＤＢＭＳ）クライアント１０６は、通信管理部１１４にコミット要求の電文領域を確保し、図１３に示すトランザクションコミット要求電文のコマンドパラメタブロック１３１１に、コミット要求であることを示すコマンド識別子を設定する。なお、コマンド識別子の後方の次ブロックオフセットには、次ブロックがあるときには、その次ブロックのオフセットが設定される。

【０１６０】次に、ステップ１２１６において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、通信管理部１１４より、サーバ１０３へ電文を送信し、トランザクションコミット要求処理を終了する。

【０１６１】なお、ステップ１２０２において、反映要求がないとして、ステップ１２０３に進んだ場合に、ステップ１２０３では、ステップ１２１５と同様にして、トランザクションコミット要求の電文を作成し、さらに、ステップ１２０４において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、通信管理部１１４より、サーバ１０３へ電文を送信し、トランザクションコミット要求処理を終了する。

【０１６２】次に、図１３を用いて、トランザクションコミット要求電文のデータ構造について説明する。通信電文１３０１の電文制御ヘッダは、（１）「電文の形式」（サーバへの送信データであることを示す値など），（２）「電文のサイズ」，（３）「先頭ブロックオフセット」（先頭のコマンドパラメタブロック１３０３の位置を示すオフセット値；通信電文１３０１の先頭からのオフセット値として与えられている），（４）「ブロック総数」（通信電文１３０１に保持するコマンドパラメタブロック１３０３の総数），（５）「変換ＯＩＤ総数」（図１２のステップ１２０５で設定）から構成されている。「ブロック総数」に設定されたデータから、通信電文１３０１が5つのコマンドパラメタブロック１３０３，１３０４，１３０５，１３０８，１３１１によって構成されていることが示されている。また、

「変換ＯＩＤ総数１３０２」により、この通信電文処理で変換ＯＩＤテーブルに登録する要素の数が”２”であることが示されている。

【０１６３】１番目のコマンドパラメタブロック１３０３には、「コマンド識別子」に”生成”が設定されており、「オブジェクトデータ」として”オブジェクトａ”が設定されており、オブジェクトａの生成要求が記されている。コマンドパラメタブロック１３０３では、「ＯＩＤ変換テーブル登録要請フラグ」がＯＮに設定されており、データベース管理システム（ＤＢＭＳ）サーバ１０３で要求を処理するときに、ＯＩＤ変換テーブル１２５にこのオブジェクトについてのエントリを登録するように指示されている。また、「次ブロックオフセット」により、コマンドパラメタブロック１３０４の先頭位置を指示している。

【０１６４】２番目のコマンドパラメタブロック１３０４には、コマンドパラメタブロック１３０３と同様に、オブジェクトｂについての生成要求が記されている。コマンドパラメタブロック１３０４の「ＯＩＤ変換テーブル登録要請フラグ」がＯＮに設定されており、また、「次ブロックオフセット」により、コマンドパラメタブロック１３０５の先頭位置を指示している。

【０１６５】３番目のコマンドパラメタブロック１３０５には、「コマンド識別子」に”更新”が設定されており、「更新対象データ」として”仮ＯＩＤａ”が設定されており、オブジェクトａの更新要求が記されている。「次ブロックオフセット」により、コマンドパラメタブロック１３０８の先頭位置を指示している。「部分更新数」には、図１１に示した部分更新情報１１０３のチェインの数が設定される。「部分更新情報オフセット」は、部分更新情報１３０６の先頭位置を指示している。「ＯＩＤ変換数」は、図１１に示したＯＩＤ変換情報１１０５のチェインの数が設定される。「ＯＩＤ変換情報オフセット」は、ＯＩＤ変換情報１３０７の先頭位置を指示している。

【０１６６】さらに、部分更新情報１３０６には、オブジェクトｂへのリンク先情報の設定情報が記されている。「部分更新オフセット」，「部分更新オフセット」及び「更新データ内容」（リンク先情報）は、図１１に示した部分更新情報１１０３の内容が設定される。

【０１６７】また、ＯＩＤ変換情報１３０７には、リンク先情報内のオブジェクトｂの仮ＯＩＤを変換するための情報が記されている。「ＯＩＤ変換位置オフセット」は、図１１に示したＯＩＤ変換情報１１０５の内容が設定される。

【０１６８】４番目のコマンドパラメタブロック１３０８，部分更新情報１３０９，ＯＩＤ変換情報１３１０には、コマンドパラメタブロック１３０５，部分更新情報１３０６，ＯＩＤ変換情報１３０７と同様にして、オブジェクトｂの更新要求が記されている。

【０１６９】５番目のコマンドパラメタブロック１３１１には、コミット要求が記されている。このコマンドパラメタブロック１３１１により、データベース管理システム（ＤＢＭＳ）サーバ１０３でトランザクションコミットが行なわれることにより、コマンドパラメタブロック１３０３からコマンドパラメタブロック１３１０による要求が、データストアへ反映されることが保証される。

【０１７０】次に、図１４を用いて、図１２のステップ１２０８に示したデータベース管理システム（ＤＢＭＳ）クライアントにおけるオブジェクト生成要求電文作成の処理の詳細について説明する。図１４は、本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト生成要求電文作成の処理手順を示すフローチャートである。

【０１７１】ステップ１４０１において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト１０８のデータサイズをもとに電文に必要な領域サイズを算出する。

【０１７２】ステップ１４０２において、通信管理部１１４は、ステップ１４０１で算出した値を元に、電文領域を確保する。

【０１７３】ステップ１４０３において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト状態管理エントリ１１０４ｂ内の「ＯＩＤ変換テーブル登録要請フラグ」がＯＮであるかを判定する。「ＯＩＤ変換テーブル登録要請フラグ」がＯＮの場合は、ステップ１４０４に進み、ＯＮでない場合には、ステップ１４０５に進む。

【０１７４】ステップ１４０４において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、コマンドパラメタ中の「ＯＩＤ変換テーブル登録要請フラグ」をＯＮに設定する。即ち、図１３に示したコマンドパラメタブロック１３０３，１３０４中の「ＯＩＤ変換テーブル登録要請フラグ」をＯＮに設定する。

【０１７５】ステップ１４０５において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト１０８のデータをコマンドパラメタ中の新規オブジェクトデータ領域にコピーする。即ち、オブジェクトデータをコマンドパラメタブロック１３０３，１３０４の中の「オブジェクトデータ」の領域に設定する。ステップ１４０５を終えた後、生成要求電文作成の処理を終了する。

【０１７６】次に、図１５を用いて、図１２のステップ１２１３に示したデータベース管理システム（ＤＢＭＳ）クライアントにおけるオブジェクト更新要求電文作成の処理の詳細について説明する。図１５は、本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト更新要求電文作成の処理手順を示すフローチャートである。

【０１７７】ステップ１５０１において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、オブジェクト状態管理エントリの部分更新情報のチェインをたどり、更新要求数をカウントし、電文のコマンドパラメタブロックに必要なサイズを算出する。即ち、図１１に示した部分更新情報１１０３のチェインをたどり、部分更新情報のエントリ数をカウントする。

【０１７８】ステップ１５０２において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ＯＩＤ変換情報チェインをたどり、変換する仮ＯＩＤ数をカウントし、電文のコマンドパラメタブロックに必要なサイズを算出する。即ち、図１１に示したＯＩＤ変換情報１１０５のチェインをたどり、ＯＩＤの変換要求数をカウントする。

【０１７９】ステップ１５０３において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ステップ１５０１およびステップ１５０２で得た値をもとに、更新要求コマンドパラメタブロック全体で必要なサイズを算出し、通信管理部１１４は、電文領域を確保する。

【０１８０】ステップ１５０４において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、更新対象オブジェクトのＯＩＤを通信電文に設定する。即ち、図１１に示したオブジェクト状態管理エントリ１１０４ｂの仮ＯＩＤｂを、図１３に示したコマンドパラメタブロック１３０５の「更新対象」に設定する。

【０１８１】ステップ１５０５において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、部分更新情報１１０３のチェインをたどり、１つの部分更新情報１１０３を取得する。

【０１８２】ステップ１５０６において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ステップ１５０５で得た部分更新情報を参照し、「部分更新位置オフセット」，「更新範囲」を取得し、これを通信電文に設定する。また、オブジェクトキャッシュ上のオブジェクトの更新内容を参照し、更新データ内容を電文に設定する。即ち、図１１に示した部分更新情報１１０３の「部分更新位置オフセット」，「部分更新範囲サイズ」を取得し、図１３に示した部分更新情報１３０６の「部分更新位置オフセット」，「部分更新範囲サイズ」に設定し、また、更新データ内容（リンク先情報）を設定する。

【０１８３】ステップ１５０７において、データベース管理システム（ＤＢＭＳ）クライアント１０６では、部分更新情報チェインで続き要素があるかを判定する。続きがある場合は、ステップ１５０５に戻り、ステップ１５０５からステップ１５０７の処理を繰り返し、すべての部分更新情報の電文を作成する。即ち、図１３に示した部分更新情報１３０６及び部分更新情報１３０９が作成される。

【０１８４】一方、ステップ１５０７の判定において、続きがない場合は、ステップ１５０８に進む。

【０１８５】ステップ１５０８において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、更新情報の数および電文中で更新情報を設定した位置のオフセット情報を、コマンドパラメタブロックのブロック制御ヘッダに設定する。即ち、図１３に示したコマンドパラメタブロック１３０５の「部分更新数」に、更新情報の数を設定し、「部分更新情報オフセット」に、電文中で更新情報を設定した位置のオフセット情報を設定する。

【０１８６】ステップ１５０９において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、更新データ中にＯＩＤ変換情報があるか否かを判定する。即ち、図１１に示したオブジェクト状態管理エントリ１１０４ｂの「ＯＩＤ変換情報アドレス」が、ＯＩＤ変換情報１１０５の先頭アドレスを保持しているか判定する。このオブジェクト内に、ＯＩＤ変換が必要であるかを判定する。

【０１８７】ＯＩＤ変換情報がある場合は、ステップ１５１０に進み、ＯＩＤ変換情報がない場合は、ステップ１５１２に進む。

【０１８８】ステップ１５１０において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ＯＩＤ変換情報チェインをたどり、１つのＯＩＤ変換情報を得る。即ち、図１１に示したＯＩＤ変換情報１１０５を取得する。

【０１８９】次に、ステップ１５１１において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ＯＩＤ変換情報を電文に設定する。ここで、ＯＩＤ変換する位置を示すオフセットを設定する。

【０１９０】即ち、図１３に示したＯＩＤ変換情報１３０７を設定する。また、このとき、「次ＯＩＤ変換オフセット」を設定する。

【０１９１】ステップ１５１２において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ＯＩＤ変換情報チェインで続き要素があるかを判定する。続きがある場合は、ステップ１５１０に戻り、ステップ１５１０からステップ１５１２の処理を繰り返す。これにより、すべてのＯＩＤ変換情報の電文が作成される。即ち、ＯＩＤ変換情報１３０７，１３１０が設定される。続きがない場合は、ステップ１５１３に進む。

【０１９２】ステップ１５１３において、データベース管理システム（ＤＢＭＳ）クライアント１０６は、ステップ１５０９からステップ１５１２の処理で作成したＯＩＤ変換情報の数，および電文中でのＯＩＤ変換情報の位置を示すオフセットをコマンドパラメタブロックのブロック制御ヘッダに設定する。即ち、図１３に示したコマンドパラメタブロック１３０５，１３０８の「ＯＩＤ変換数」に、ＯＩＤ変換情報の数を設定し、「ＯＩＤ変

換情報オフセット」に、電文中でのＯＩＤ変換情報の位置を示すオフセットを設定する。

【０１９３】ステップ１５１３を終えた後、当該更新要求電文作成処理を終了する。

【０１９４】次に、図１６を用いて、図２のステップ２０６に示したデータベース管理システム（ＤＢＭＳ）サーバにおけるトランザクションコミットの処理の詳細について説明する。図１６は、本発明の一実施形態によるオブジェクト管理方法におけるトランザクションコミットの処理手順を示すフローチャートである。

【０１９５】ステップ１６０１において、サーバアプリケーション１１８ａは、通信管理部１０７から、電文のアドレスを取得する。

【０１９６】ステップ１６０２において、１２４は、通信電文中の変換ＯＩＤ総数（図１３に示した変換ＯＩＤ総数１３０２）をもとに、総数分の要素配列を保持可能な領域を確保し、ＯＩＤ変換テーブル１２５を初期化する。

【０１９７】ステップ１６０３において、サーバアプリケーション１１８は、ステップ１６０１で得た通信電文から、電文制御ヘッダの「先頭ブロックオフセット」を参照して、コマンドパラメタブロック１３０３を取得する。

【０１９８】ステップ１６０４において、サーバアプリケーション１１８は、コマンドパラメタブロック１３０３の「コマンド識別子」が「生成」を示す値であるかを判定する。「生成」の場合は、ステップ１６０５に進む。ステップ１６０５において、オブジェクト管理部１１９は、生成要求の電文解析処理を行なう。生成要求の電文解析処理の詳細については、図１７を用いて後述する。ステップ１６０５を終えた後、ステップ１６１０に進む。

【０１９９】ステップ１６１０において、次のコマンドパラメタブロックを取得する。図１３に示すように、コマンドパラメタブロック１３０３の「コマンド識別子」は「生成」であり、続くコマンドパラメタブロック１３０４の「コマンド識別子」も「生成」であるので、ステップ１６０５による生成コマンド処理を実行する。

【０２００】一方、ステップ１６０４において、「生成」でない場合は、ステップ１６０６へ行き、コマンド識別子が「更新」を示す値であるかを判定する。

【０２０１】図１３に示したコマンドパラメタブロック１３０５，１３０８の「コマンド識別子」は「更新」であるので、ステップ１６０７に進み、オブジェクト管理部１１９は、更新要求の電文解析処理を行なう。更新要求の電文解析処理の詳細については、図１８を用いて後述する。ステップ１６０７を終えた後、ステップ１６１０に進む。

【０２０２】一方、ステップ１６０６で「更新」でない場合は、ステップ１６０８へ行き、コマンド識別子が「コミット」を示す値であるかを判定する。

【０２０３】図１３に示したコマンドパラメタブロック１３１１の「コマンド識別子」は「コミット」であるので、ステップ１６０９に進み、サーバアプリケーション１１８は、トランザクション管理部１１７へトランザクションコミットの要求を行なう。ステップ１６０９を終えた後、当該コミット処理を終了する。

【０２０４】一方、ステップ１６０８において、要求種別が「コミット」でない場合は、ステップ１６１０に進み、ステップ１６１０では、電文中の次のコマンドパラメタブロックを取得し、ステップ１６０４に戻る。以上の繰り返しにより、「コミット」要求までのすべての「生成」および「更新」の反映要求に対する処理が行なわれる。

【０２０５】次に、図１７，図１９を用いて、図１６のステップ１６０５に示したデータベース管理システム（ＤＢＭＳ）サーバにおけるオブジェクト生成コマンド処理の詳細について説明する。図１７は、本発明の一実施形態によるオブジェクト管理方法における生成コマンドの処理手順を示すフローチャートであり、図１９は、本発明の一実施形態によるオブジェクト管理方法におけるＤＢＭＳサーバ内でのオブジェクトデータの構造の説明図である。

【０２０６】ステップ１７０１において、オブジェクト管理部１１９は、コマンドパラメタブロック中のオブジェクトデータサイズを取得し、オブジェクトデータサイズのオブジェクトを保持できるように、オブジェクトキャッシュ領域を確保する。

【０２０７】即ち、図１３に示したコマンドパラメタブロック１３０３，１３０４の「オブジェクトデータ」の構成は、図８に示したような構成になっており、図８に示したオブジェクトデータ１０８のオブジェクト制御ヘッダ８０１の中の「オブジェクトの全体のサイズ」からオブジェクトデータサイズを取得する。

【０２０８】ステップ１７０２において、オブジェクト管理部１１９は、コマンドパラメタブロック中のオブジェクトデータを、ステップ１７０１で確保した領域にコピーする。即ち、図１３に示したコマンドパラメタブロック１３０３，１３０４の「オブジェクトデータ」を、図１９に示したオブジェクトキャッシュ１２０のオブジェクト１９０１ａにコピーする。

【０２０９】ステップ１７０３において、オブジェクト管理部１１９は、格納制御部１２６にオブジェクト格納を要求する。格納位置が確定することにより、永続識別子（ＰＩＤ）が与えられ、ＯＩＤ管理部１２４よりこのオブジェクトに対して正式ＯＩＤが割り付けられる。ここで、割り付けられた正式ＯＩＤをオブジェクトキャッシュ１２０上のオブジェクト１９０１ａの制御ヘッダの自分自身のＯＩＤの保持域（図１９における正式ＯＩＤａの領域）に設定する。即ち、図１９のオブジェクトキ

ャッシュ１２０上のオブジェクト１９０１ａの「自分自身のＯＩＤ」に「正式ＯＩＤａ」が設定され、オブジェクト１９０１ｂの「自分自身のＯＩＤ」に「正式ＯＩＤｂ」が設定される。

【０２１０】ステップ１７０４において、オブジェクト管理部１１９は、このオブジェクトデータに対して、オブジェクト状態管理エントリを割り付け、初期化する。これにより、このオブジェクトが活性化状態となる。

【０２１１】ステップ１７０５において、オブジェクト管理部１１９は、コマンドパラメタブロック１３０３，１３０４中の「ＯＩＤ変換テーブル登録要請フラグ」がＯＮであるかを判定する。フラグがＯＮの場合は、ステップ１７０６に進む。

【０２１２】ステップ１７０６において、オブジェクト管理部１１９は、オブジェクトの仮ＯＩＤ内の通番を元に、ＯＩＤ変換テーブルに登録する。即ち、図１３に示したコマンドパラメタブロック１３０３，１３０４の中の構造は、図８に示したとおりであり、図８に示したオブジェクト１０８のオブジェクト制御ヘッダ８０１の中の「自分自身のＯＩＤ」の中の仮ＯＩＤ８０３内の生成通番をもとに、図１９に示したＯＩＤ変換テーブル１２５において、このオブジェクトに関するエントリの登録位置を確定し、活性化されているオブジェクトデータのアドレスを設定する。

【０２１３】ステップ１７０６を終えた後、または、ステップ１７０５でフラグがＯＮでない場合は、生成コマンド処理を終了する。

【０２１４】次に、図１８を用いて、図１６のステップ１６０７に示したデータベース管理システム（ＤＢＭＳ）サーバにおけるオブジェクト更新コマンド処理の詳細について説明する。図１８は、本発明の一実施形態によるオブジェクト管理方法における更新コマンドの処理手順を示すフローチャートである。

【０２１５】ステップ１８０１において、オブジェクト管理部１１９は、更新対象オブジェクトをオブジェクトキャッシュに活性化する。ここで、更新対象が新規オブジェクトの場合は、先に、図１７で説明した処理により、すでに活性化状態になっており、ＯＩＤ変換テーブルを参照することにより、対象オブジェクトを取得することができる。また、更新対象がデータストアに格納されている既存のオブジェクトの場合は、通信電文中に指定される正式ＯＩＤにより、周知の活性化機構を用いて、データストアから活性化されるものとする。

【０２１６】ステップ１８０２において、オブジェクト管理部１１９は、ブロック制御ヘッダを参照して、ブロック中の１つの部分更新情報を取得する。即ち、図１３に示したコマンドパラメタブロック１３０５の「部分更新情報オフセット」から部分更新情報１３０６を取得する。

【０２１７】ステップ１８０３において、オブジェクト管理部１１９は、ステップ１８０２で取得した部分更新情報をもとに、ステップ１８０１で取得したオブジェクトデータを書き換える。即ち、ステップ１８０１において、初期化した領域に部分更新情報を入れる。

【０２１８】ステップ１８０４において、オブジェクト管理部１１９は、更新情報のブロック制御ヘッダを参照し、先のステップ１８０３で処理した部分更新情報に続きがあるか判定する。

【０２１９】続きがある場合は、ステップ１８０２に戻り、すべての部分更新が行なわれるようにする。即ち、図１３に示したコマンドパラメタブロック１３０８の「部分更新情報オフセット」から部分更新情報１３０９を取得し、取得した部分更新情報をもとに、ステップ１８０１で取得したオブジェクトデータを書き換える。続きがない場合は、ステップ１８０５に進み。

【０２２０】ステップ１８０５において、オブジェクト管理部１１９は、ブロック制御ヘッダを参照し、ブロック中にＯＩＤ変換情報を保持しているか判定する。即ち、図１３に示したコマンドパラメタブロック１３０５の「ＯＩＤ変換情報オフセット」からＯＩＤ変換情報を保持しているか判定する。保持していない場合は、ステップ１８１２に進み、保持している場合は、ステップ１８０６に進む。

【０２２１】ステップ１８０６において、オブジェクト管理部１１９は、ブロック中から１つのＯＩＤ変換情報を取得する。即ち、図１３に示したＯＩＤ変換情報１３０７を取得する。

【０２２２】ステップ１８０７において、オブジェクト管理部１１９は、ステップ１８０６で取得したＯＩＤ変換情報内のオフセットをもとに、オブジェクトデータ内に埋め込まれている仮ＯＩＤを取得する。即ち、図１３に示したコマンドパラメタブロック１３０５の中の「更新対象」の「仮ＯＩＤａ」を取得する。

【０２２３】ステップ１８０８において、オブジェクト管理部１１９は、ステップ１８０７で取得した仮ＯＩＤの内容を参照し、仮ＯＩＤの内部データに保持されている生成通番を取得する。即ち、仮ＯＩＤの構造は、図８に示したようであり、図８の仮ＯＩＤ８０３の「生成通番」を取得する。

【０２２４】ステップ１８０９において、オブジェクト管理部１１９は、ステップ１８０８で得られた値をもとに、この値を、ＯＩＤ変換テーブル配列の要素位置として、オブジェクトに対応するエントリを取得する。このエントリには、図１９に示したように、オブジェクトキャッシュ１２０上に活性化されたオブジェクトのアドレスが設定されている。オブジェクト１９０１の制御ヘッダには、自分自身の正式ＯＩＤが設定されているので、これを参照することにより、正式ＯＩＤを取得する。

【０２２５】ステップ１８１０において、オブジェクト管理部１１９は、オブジェクトデータ内の仮ＯＩＤを、

ステップ１８０９で得られた正式ＯＩＤに書き換える。即ち、図１９に示したオブジェクト１９０１ａの中で、図８に示したオブジェクト制御ヘッダ８０１の中の「リンク先ＯＩＤ」に正式ＯＩＤｂを書き込む。

【０２２６】ステップ１８１１において、オブジェクト管理部１１９は、ブロック中に続きのＯＩＤ変換情報があるか判定する。ＯＩＤ変換情報がある場合は、ステップ１８０６に戻り、すべてのＯＩＤ変換を処理するまで、ステップ１８０６からステップ１８１１の処理を繰り返す。これにより、図１９に示したオブジェクト１９０１ｂの中の「リンク先ＯＩＤ」に正式ＯＩＤａを書き込まれる。

【０２２７】一方、ステップ１８１１において、ＯＩＤ変換情報がない場合は、ステップ１８１２に進む。

【０２２８】ステップ１８１２において、オブジェクト管理部１１９は、格納制御部１２６に対して、これまでの更新をデータストアに反映するよう要求する。ステップ１８１２を終えた後、当該更新コマンド処理を終了する。

【０２２９】ここで、図１９を用いて、データベース管理システム（ＤＢＭＳ）サーバにおけるオブジェクトデータ構造について説明する。

【０２３０】オブジェクトａ１９０１ａとオブジェクトｂ１９０１ｂが、オブジェクトキャッシュ１２０上に活性化されている。ＯＩＤ管理部１２４のＯＩＤ変換テーブル１２５には、オブジェクトａとオブジェクトｂのそれぞれのエントリが登録されている。ここで、オブジェクトａは、生成通番でａ番目に生成されたものとし、生成順に従って、ＯＩＤ変換テーブル１２５の配列でａ番目にエントリが登録されている。同様に、オブジェクトｂについても、ＯＩＤ変換テーブル１２５の配列でｂ番目にエントリが登録されている。

【０２３１】図１７のステップ１７０３により、オブジェクトａ１９０１ａとオブジェクトｂ１９０１ｂのオブジェクト制御ヘッダで、「自分自身のＯＩＤ」の保持領域に、それぞれ、「正式ＯＩＤａ」、「正式ＯＩＤｂ」が設定されている。

【０２３２】また、図１８のステップ１８１０により、オブジェクトａ１９０１ａとオブジェクトｂ１９０１ｂのリンク先情報で、「リンク先ＯＩＤ」に、それぞれ、「正式ＯＩＤｂ」、「正式ＯＩＤａ」が設定されている。

【０２３３】以上、本発明の一実施形態について説明したが、これは例示したものであり、これによって本発明が制限されるものではない。

【０２３４】なお、上述した例において、オブジェクト間の関連の多重度は１対１としていたが、多対多の関連についても、本発明を適用することができる。多対多関連の実装方法としては、一般に、直にリンク先ＯＩＤを保持するのではなく、ＯＩＤの集合を管理するコレクション機能を介して実装され、コレクションの要素のＯＩＤがリンク先オブジェクトのＯＩＤであると解釈される。この場合、基本的にはＯＩＤを扱うので、上記の方式で仮ＯＩＤを正式ＯＩＤに変換すれば、実装することが可能である。

【０２３５】また、上述した例において、生成通番の一意性を保証する期間単位をトランザクションとしているが、サーバと通信を行なうタイミングや、データベース（ＤＢ）接続、クライアントプロセス単位など、クライアントアプリケーションの形態に応じて適当に単位を設定することも考えられる。

【０２３６】本実施形態によれば、クライアントーサーバ構成のオブジェクト指向データベースシステムにおいて、新規オブジェクトに対して生成順通番をもとにした仮ＯＩＤを用いるにより、正式ＯＩＤを必要とするようなクライアントアプリケーションでのオブジェクト操作を、データベース管理システム（ＤＢＭＳ）サーバとの通信を行なうことなく（通信回数を削減して）、効率良く処理することが可能となる。

【０２３７】また、複数個の仮ＯＩＤを一括してクライアントからサーバに送信することにより、通信回数を削減することができる。

【０２３８】また、反映要求の電文にＯＩＤ変換情報を付加しておくことにより、仮ＯＩＤから正式ＯＩＤへの変換を効率よく行えるようになる。

【０２３９】

【発明の効果】本発明によれば、クライアントーサーバ型で構成されるオブジェクト指向データベースシステムにおいて、システム内でのオブジェクト識別性を保証しつつ、サーバと通信を行うことなく、クライアントアプリケーションでのオブジェクト操作を効率よく処理できる。

【図面の簡単な説明】

【図１】本発明の一実施形態によるオブジェクト管理方法を適用したオブジェクト指向データベース（ＯＯＤＢ）システムの構成図である。

【図２】本発明の一実施形態によるオブジェクト管理方法におけるクライアントアプリケーションでのアプリケーションプログラムの処理フローの概念を示すフローチャートである。

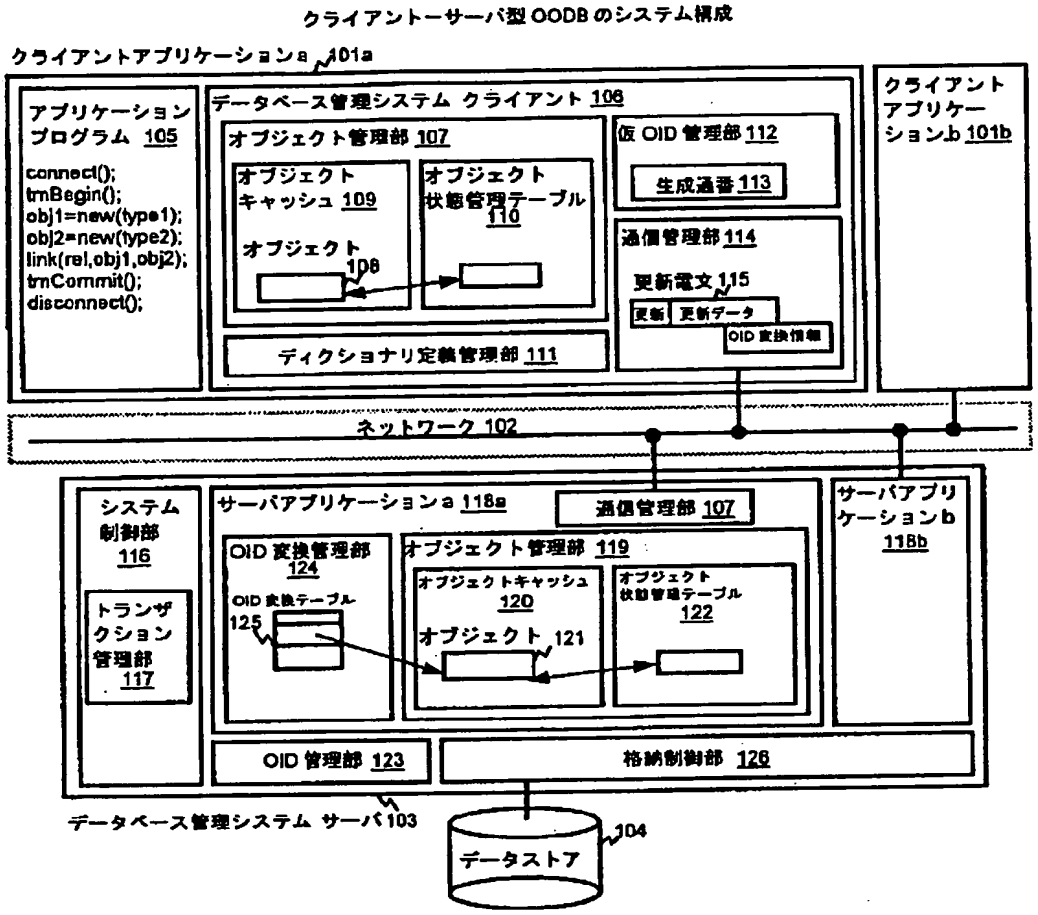
【図３】本発明の一実施形態によるオブジェクト管理方法におけるデータベース（ＤＢ）接続ＡＰＩの処理手順を示すフローチャートである。

【図４】本発明の一実施形態によるオブジェクト管理方法において用いるデータベース管理システム（ＤＢＭＳ）クライアントで取り扱う制御データの構造の説明図である。

【図５】本発明の一実施形態によるオブジェクト管理方法においてクライアントアプリケーションが作成する通信電文のデータ構造の説明図である。

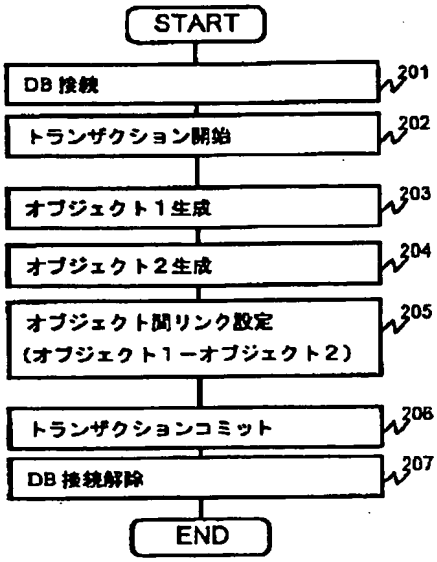
【図６】本発明の一実施形態によるオブジェクト管理方法におけるトランザクション開始ＡＰＩ関数コールの処理手順を示すフローチャートである。
【図７】本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト生成ＡＰＩの処理手順を示すフローチャートであり、
【図８】図７に示したオブジェクト生成ＡＰＩの処理によって生成されるデータベース管理システム（ＤＢＭＳ）クライアントのオブジェクトデータ構造の説明図である。
【図９】本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト間リンク設定ＡＰＩの処理手順を示すフローチャートである。
【図１０】本発明の一実施形態によるオブジェクト管理方法におけるリンク設定時の更新情報作成の処理手順を示すフローチャートである。
【図１１】本発明の一実施形態によるオブジェクト管理方法におけるリンク設定時の更新情報作成処理で扱われるデータ構造の説明図である。
【図１２】本発明の一実施形態によるオブジェクト管理方法におけるトランザクションコミットＡＰＩの処理手順を示すフローチャートである。
【図１３】本発明の一実施形態によるオブジェクト管理方法におけるトランザクションコミット要求の通信電文データ構造の説明図である。
【図１４】本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト生成要求電文作成の処理手順を示すフローチャートである。
【図１５】本発明の一実施形態によるオブジェクト管理方法におけるオブジェクト更新要求電文作成の処理手順を示すフローチャートである。
【図１６】本発明の一実施形態によるオブジェクト管理方法におけるトランザクションコミットの処理手順を示すフローチャートである。
【図１７】本発明の一実施形態によるオブジェクト管理方法における生成コマンドの処理手順を示すフローチャートである。
【図１８】本発明の一実施形態によるオブジェクト管理方法における更新コマンドの処理手順を示すフローチャートである。
【図１９】本発明の一実施形態によるオブジェクト管理方法におけるＤＢＭＳサーバ内でのオブジェクトデータの構造の説明図である。
【符号の説明】
１０１ａ…クライアントアプリケーションａ
１０１ｂ…クライアントアプリケーションｂ
１０２…ネットワーク
１０３…データベース管理システム　サーバ
１０４…データストア
１０５…アプリケーションプログラム
１０６…データベース管理システムクライアント
１０７…オブジェクト管理部
１０８…オブジェクト
１０９…オブジェクトキャッシュ
１１０…オブジェクト状態管理テーブル
１１１…ディクショナリ定義管理部
１１２…仮ＯＩＤ管理部
１１３…生成通番
１１４…通信管理部
１１５…更新電文
１１６…システム制御部
１１７…トランザクション管理部
１１８ａ…サーバアプリケーションａ
１１８ｂ…サーバアプリケーションｂ
１１９…オブジェクト管理部
１２０…オブジェクトキャッシュ
１２１…オブジェクト
１２２…オブジェクト状態管理テーブル
１２３…ＯＩＤ管理部
１２４…ＯＩＤ変換管理部
１２５…ＯＩＤ変換テーブル
１２６…格納制御部

【図１】



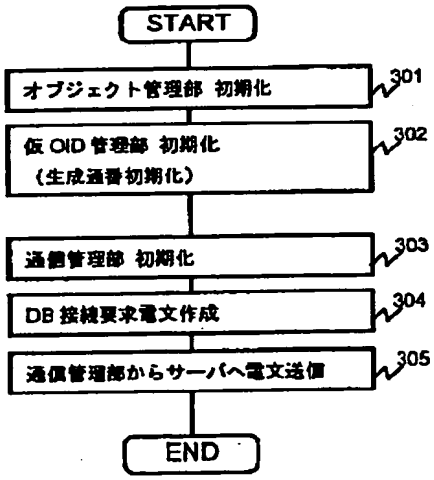
【図２】

アプリケーションプログラムの処理フロー



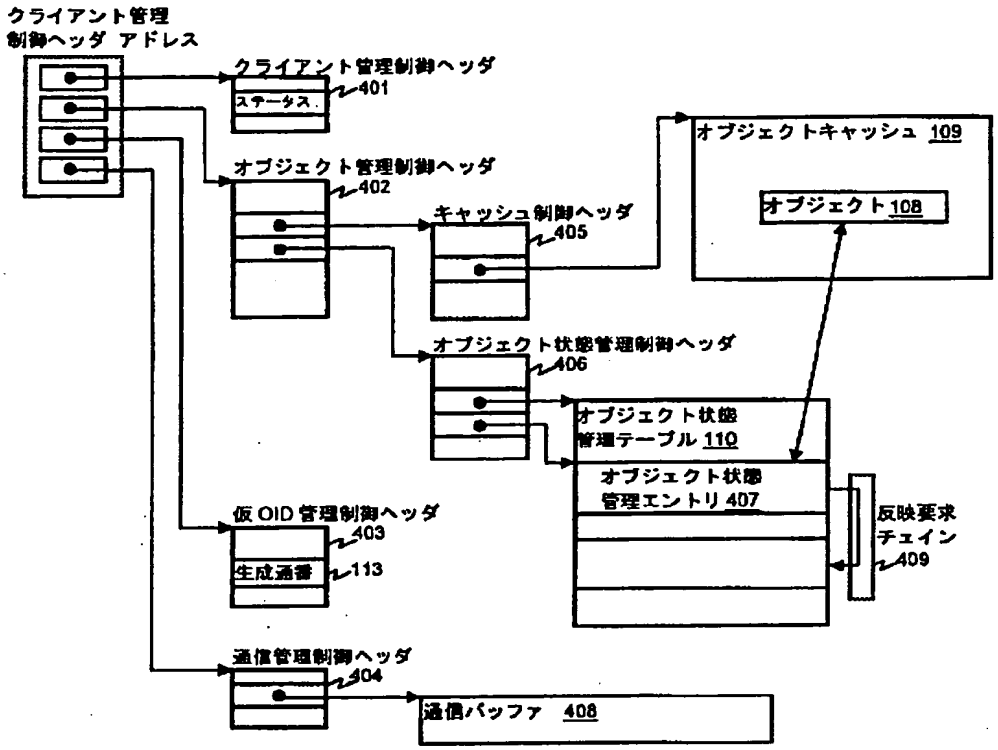
【図３】

DBMS クライアント DB 接続 API の処理フロー



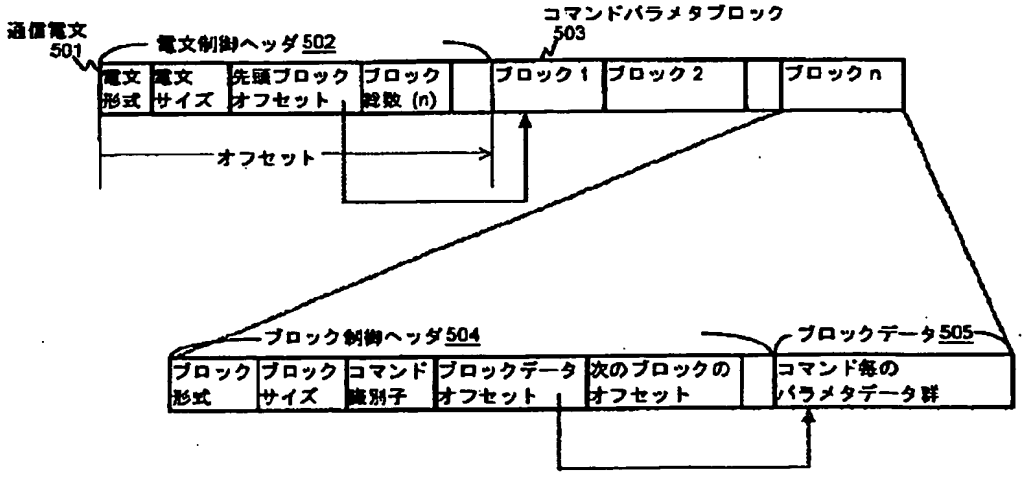
【図4】

DBMS クライアントの制御データ構造



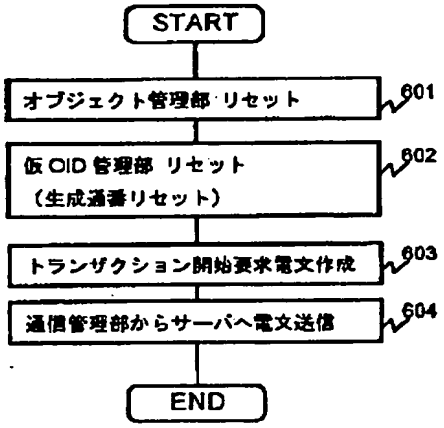
【図5】

通信電文データ構造



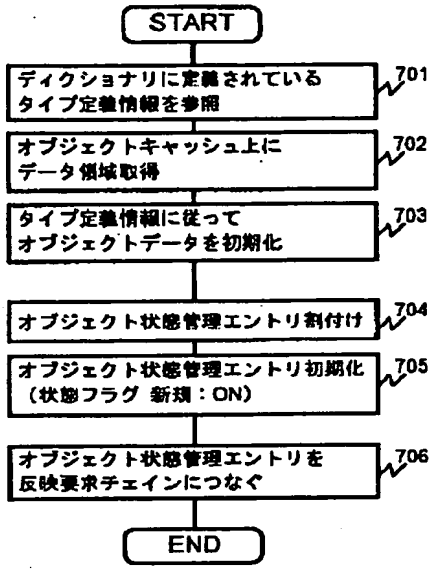
【図６】

DBMS クライアント トランザクション開始 API の処理フロー



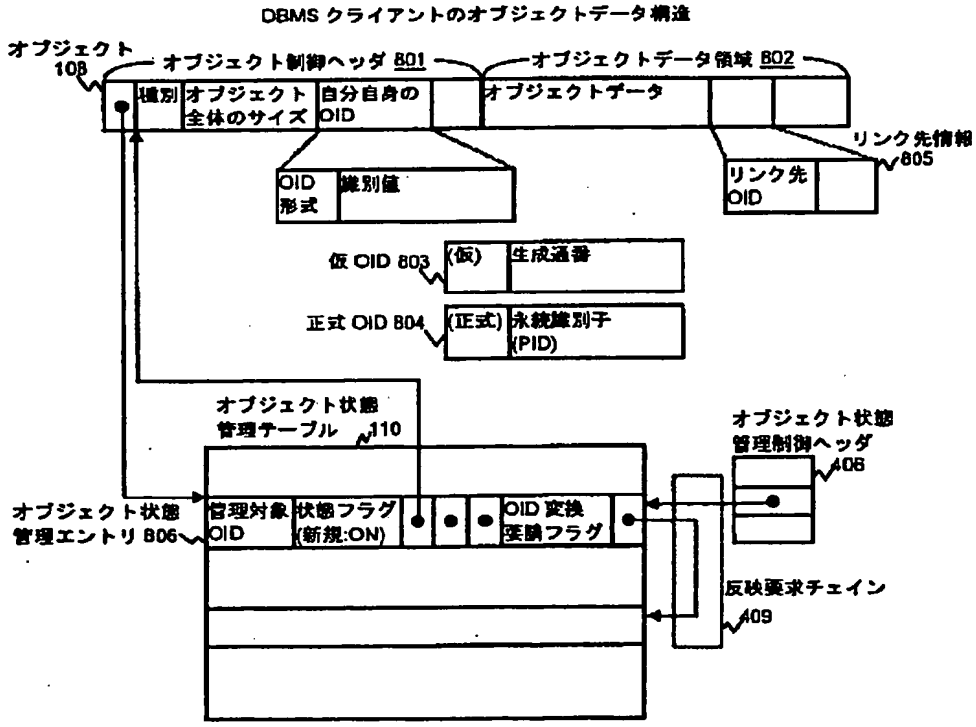
【図７】

DBMS クライアント オブジェクト生成 API の処理フロー

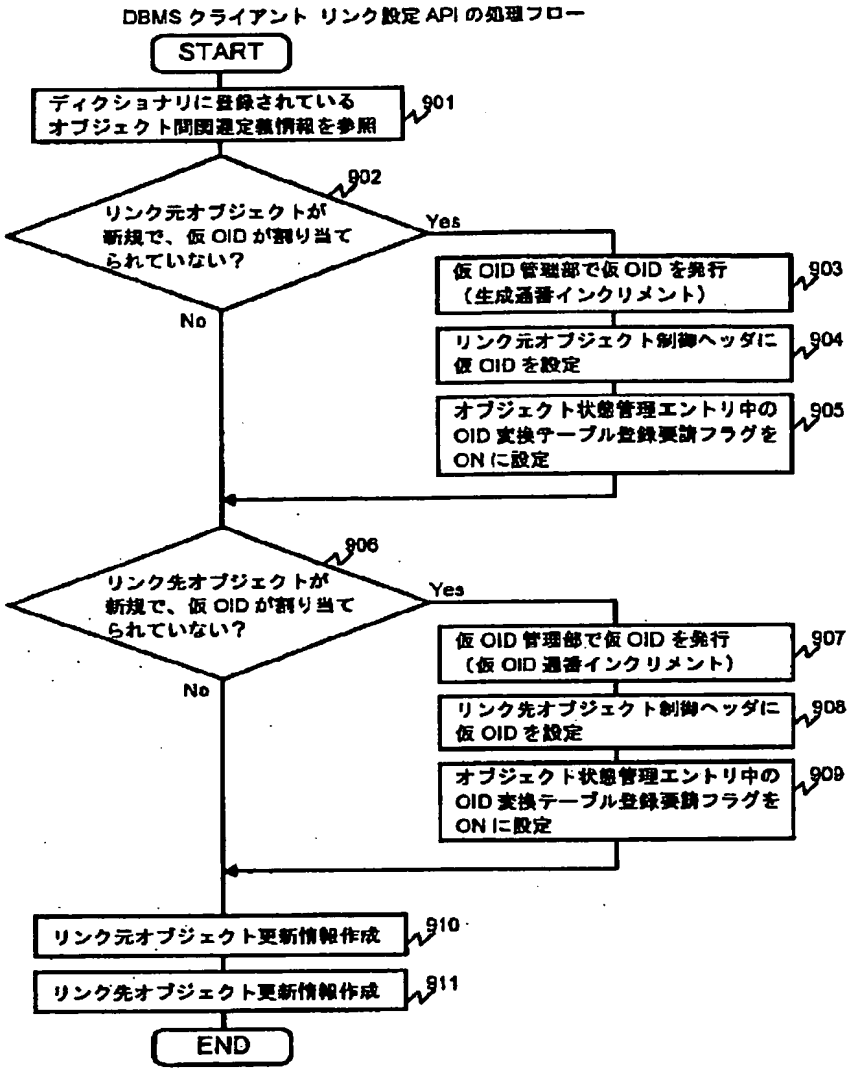


【図８】

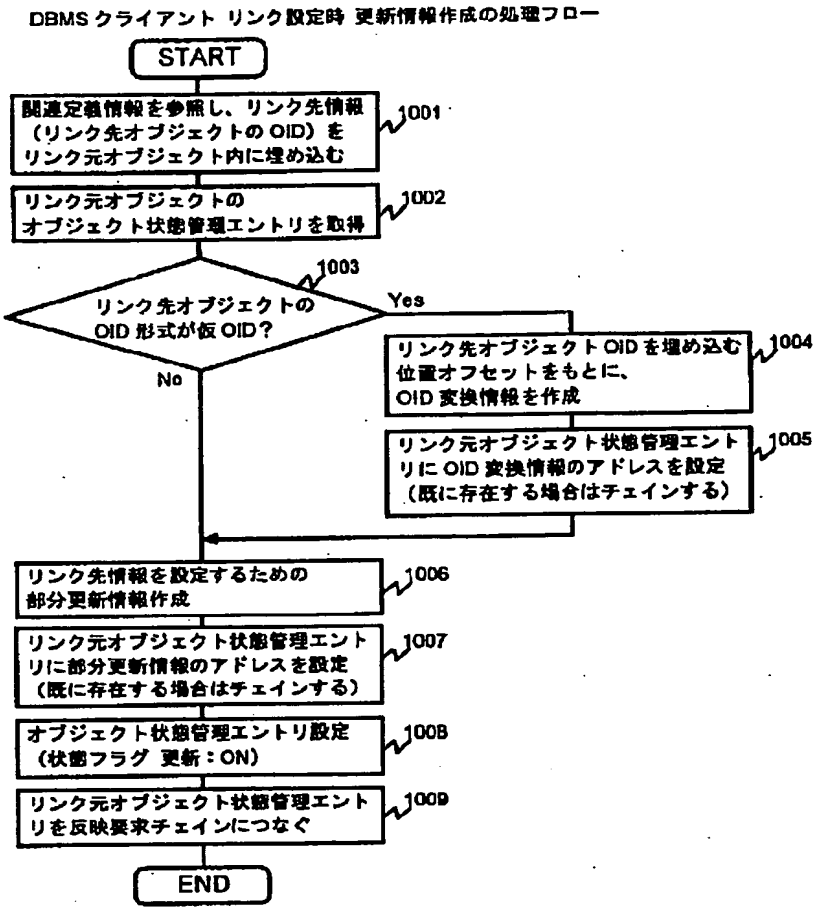
DBMS クライアントのオブジェクトデータ構造



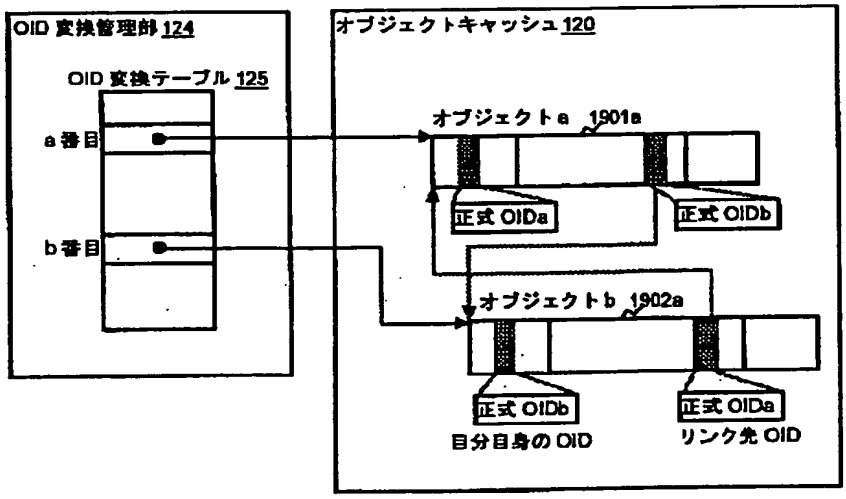
【図９】


DBMS クライアント リンク設定 API の処理フロー
START
ディクショナリに登録されている
オブジェクト間関連定義情報を参照
901
902
リンク元オブジェクトが
新規で、仮 OID が割り当て
られていない？
Yes
No
仮 OID 管理部で仮 OID を発行
（生成通番インクリメント）
903
リンク元オブジェクト制御ヘッダに
仮 OID を設定
904
オブジェクト状態管理エントリ中の
OID 変換テーブル登録要請フラグを
ON に設定
905
906
リンク先オブジェクトが
新規で、仮 OID が割り当て
られていない？
Yes
No
仮 OID 管理部で仮 OID を発行
（仮 OID 通番インクリメント）
907
リンク先オブジェクト制御ヘッダに
仮 OID を設定
908
オブジェクト状態管理エントリ中の
OID 変換テーブル登録要請フラグを
ON に設定
909
リンク元オブジェクト更新情報作成
910
リンク先オブジェクト更新情報作成
911
END

【図１０】

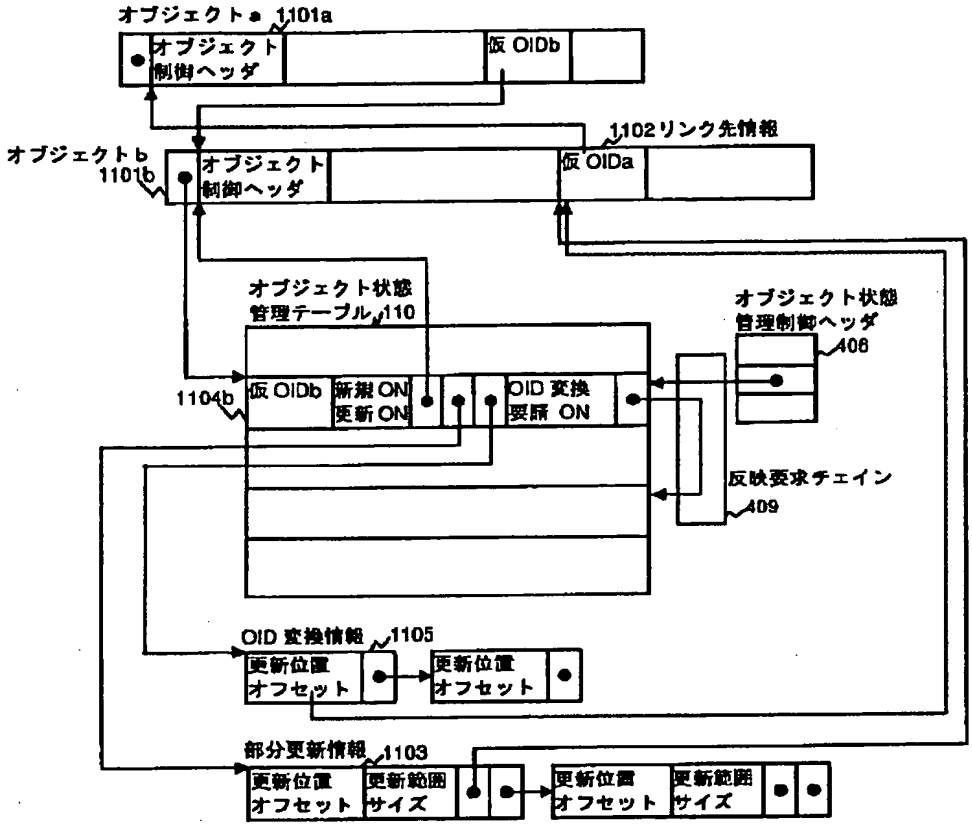

DBMS クライアント リンク設定時 更新情報作成の処理フロー
START
関連定義情報を参照し、リンク先情報（リンク先オブジェクトのOID）をリンク元オブジェクト内に埋め込む
1001
リンク元オブジェクトのオブジェクト状態管理エントリを取得
1002
1003
リンク先オブジェクトのOID形式が仮OID?
Yes
No
リンク先オブジェクトOIDを埋め込む位置オフセットをもとに、OID変換情報を作成
1004
リンク元オブジェクト状態管理エントリにOID変換情報のアドレスを設定（既に存在する場合はチェインする）
1005
リンク先情報を設定するための部分更新情報作成
1006
リンク元オブジェクト状態管理エントリに部分更新情報のアドレスを設定（既に存在する場合はチェインする）
1007
オブジェクト状態管理エントリ設定（状態フラグ 更新：ON）
1008
リンク元オブジェクト状態管理エントリを反映要求チェインにつなぐ
1009
END

【図１９】

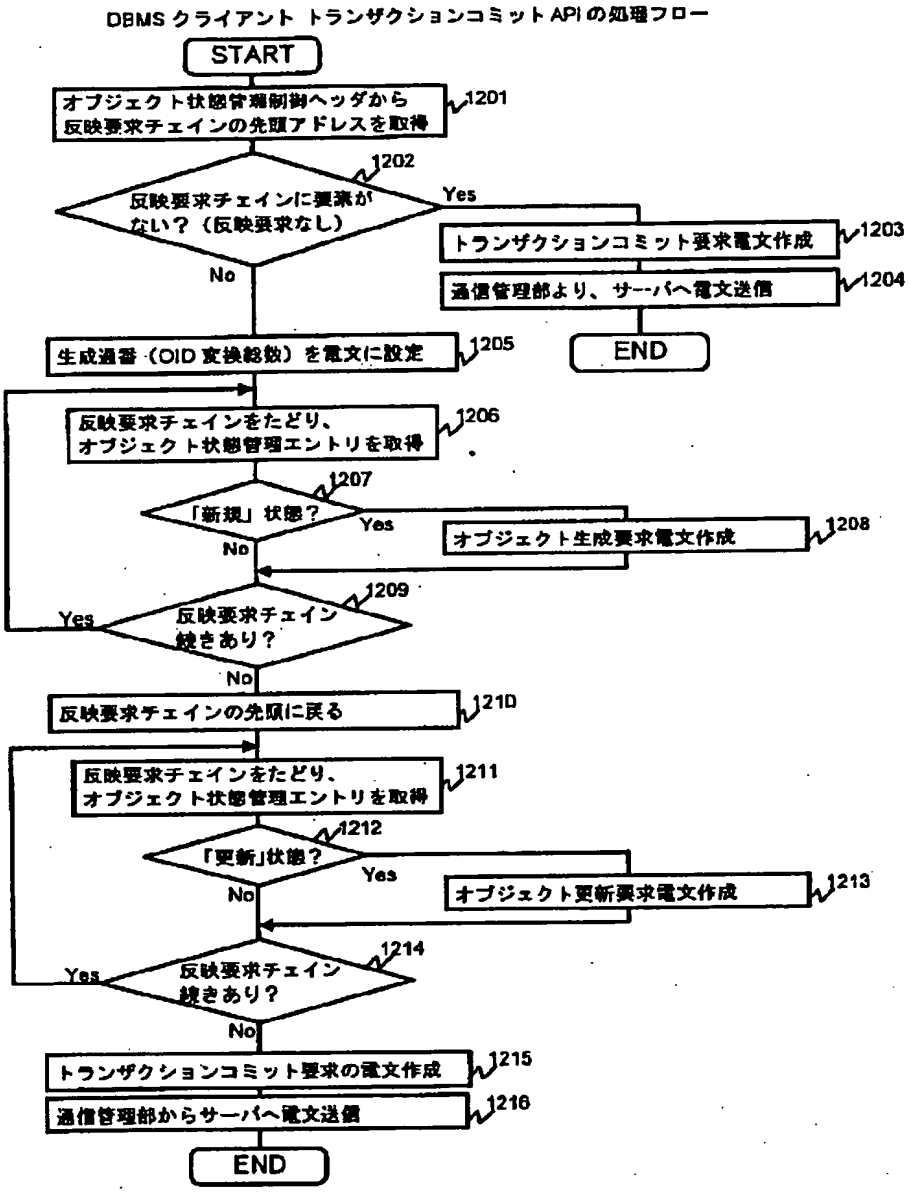

DBMS サーバのオブジェクトデータ構造
OID 変換管理部 124
OID 変換テーブル 125
a番目
b番目
オブジェクトキャッシュ 120
オブジェクトa 1901a
正式 OIDa
正式 OIDb
オブジェクトb 1902a
正式 OIDb
自分自身の OID
正式 OIDa
リンク先 OID

【図１１】

DBMS クライアント リンク設定のデータ構造

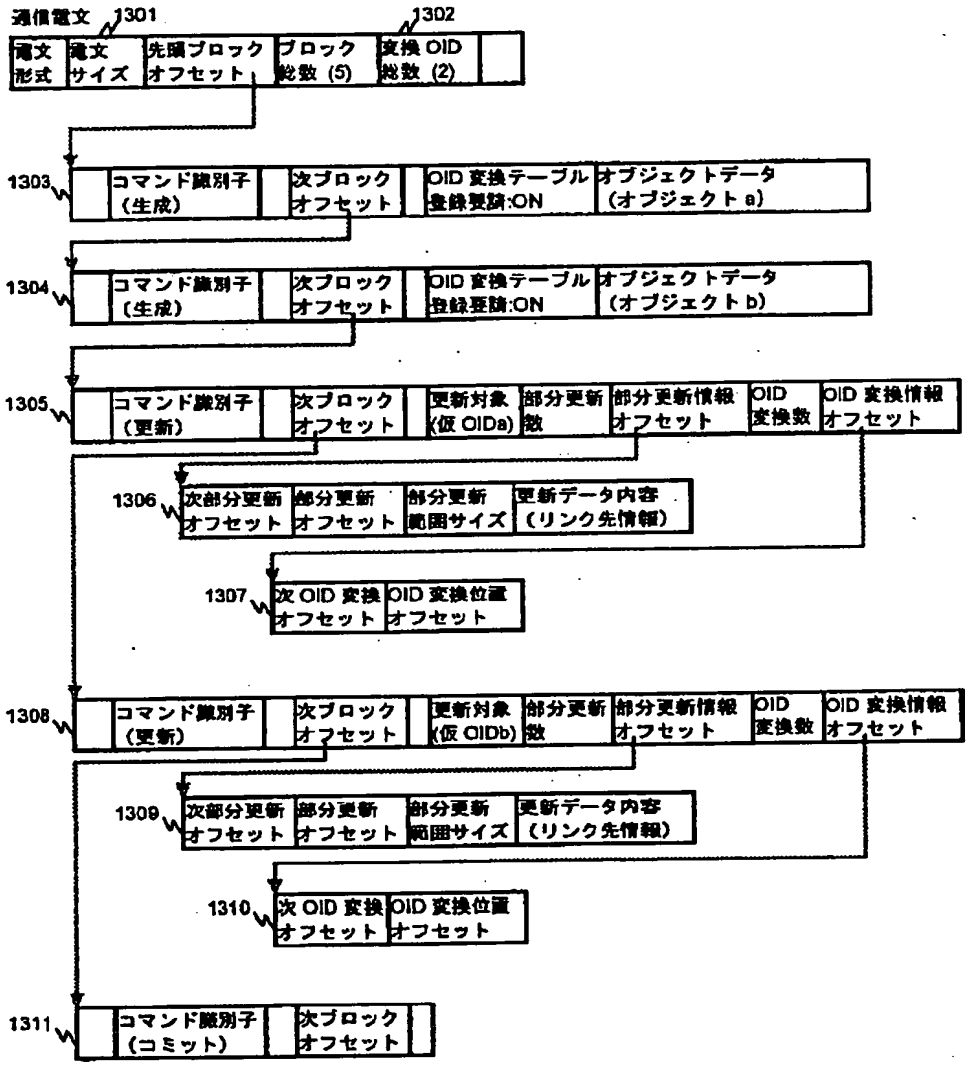


【図１２】


DBMS クライアント トランザクションコミット API の処理フロー
START
オブジェクト状態管理制御ヘッダから反映要求チェインの先頭アドレスを取得
1201
反映要求チェインに要素がない？（反映要求なし）
1202
Yes
No
トランザクションコミット要求電文作成
1203
通信管理部より、サーバへ電文送信
1204
END
生成通番（OID 変換総数）を電文に設定
1205
反映要求チェインをたどり、オブジェクト状態管理エントリを取得
1206
「新規」状態？
1207
Yes
No
オブジェクト生成要求電文作成
1208
反映要求チェイン続きあり？
1209
Yes
No
反映要求チェインの先頭に戻る
1210
反映要求チェインをたどり、オブジェクト状態管理エントリを取得
1211
「更新」状態？
1212
Yes
No
オブジェクト更新要求電文作成
1213
反映要求チェイン続きあり？
1214
Yes
No
トランザクションコミット要求の電文作成
1215
通信管理部からサーバへ電文送信
1216
END

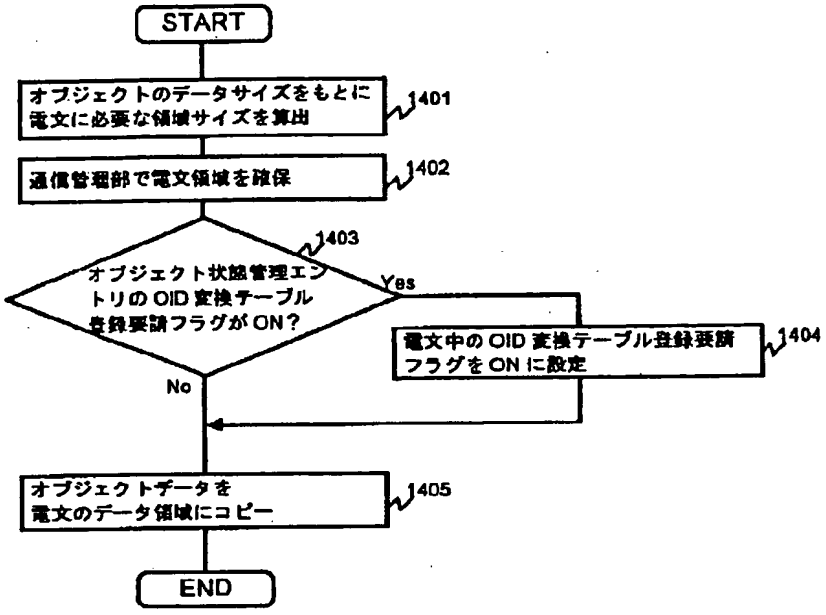
【図１３】

DBMS クライアント トランザクションコミット要求電文データ構造



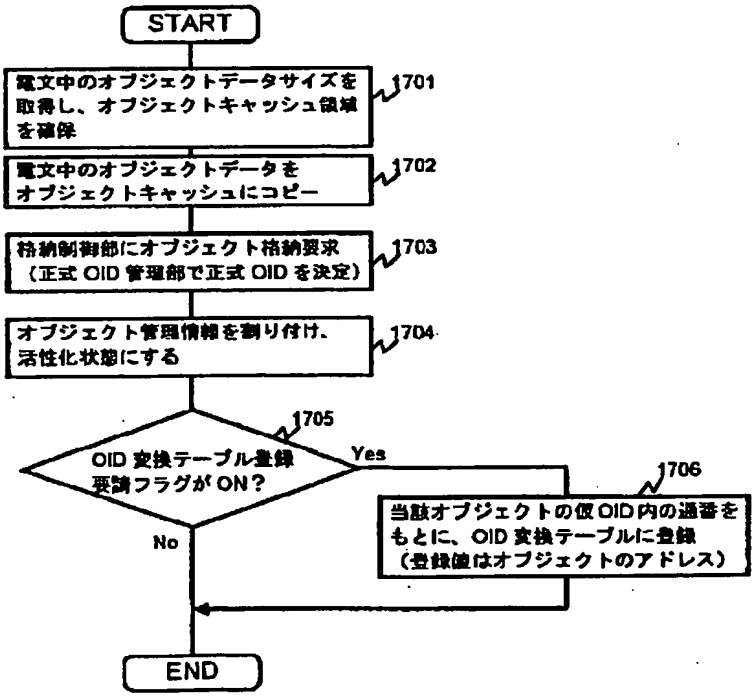
【図１４】

DBMSクライアント オブジェクト生成要求電文作成の処理フロー

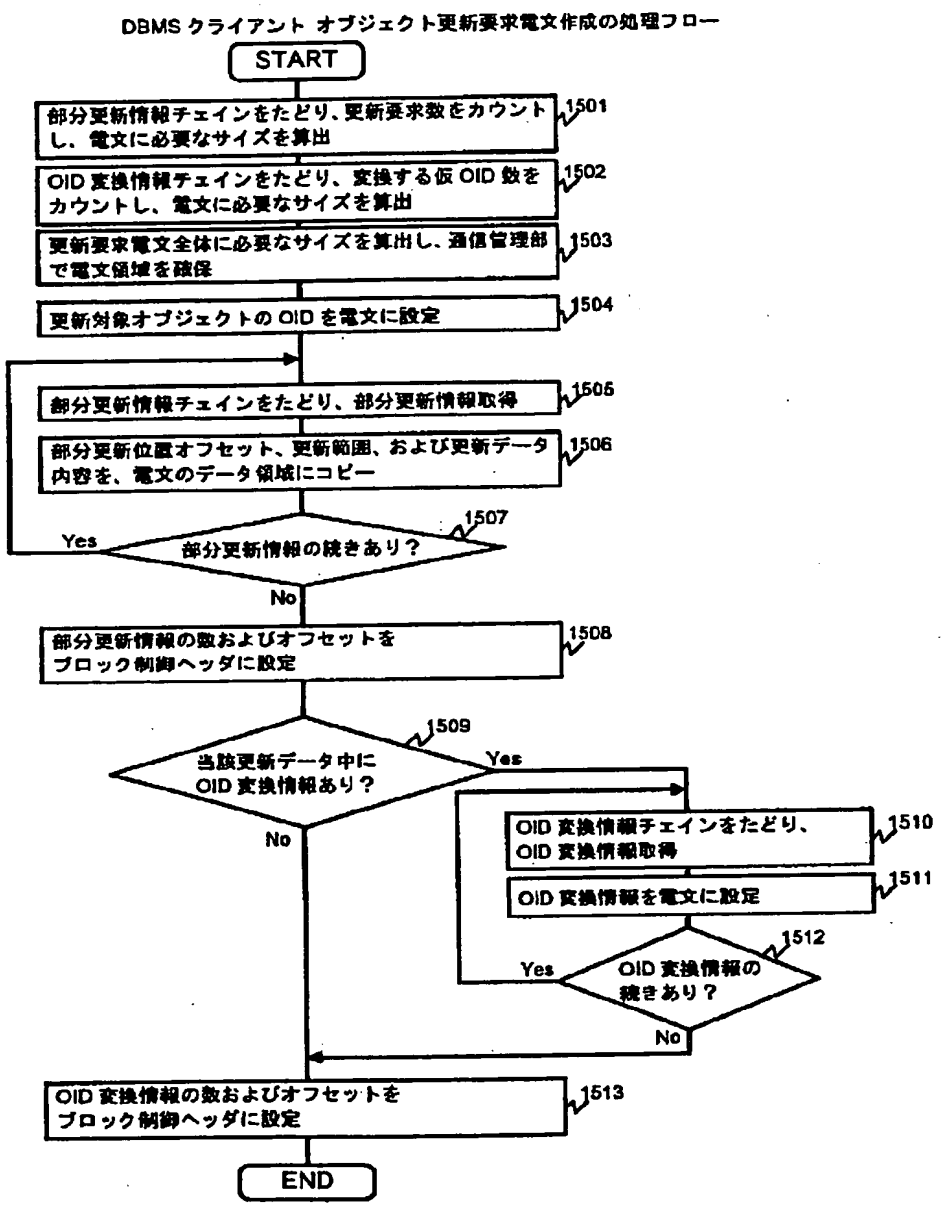


【図１７】

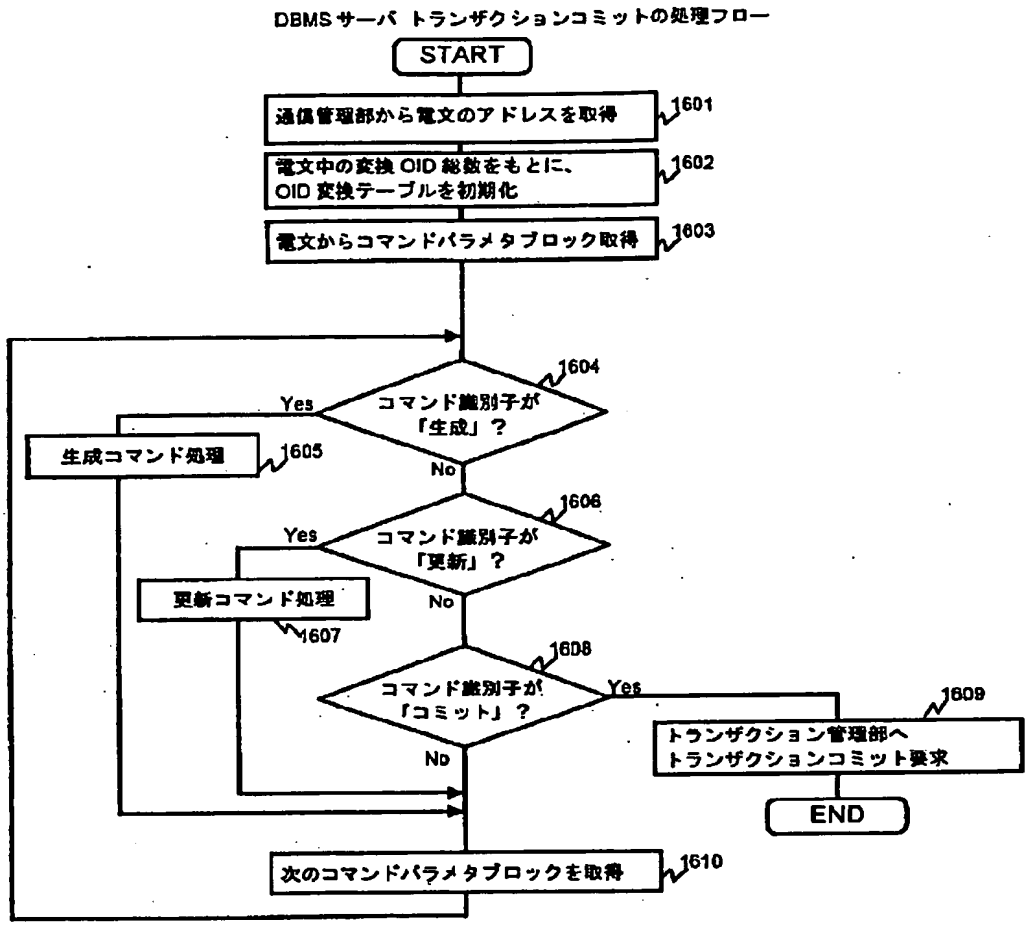
DBMSサーバ オブジェクト生成コマンド処理フロー



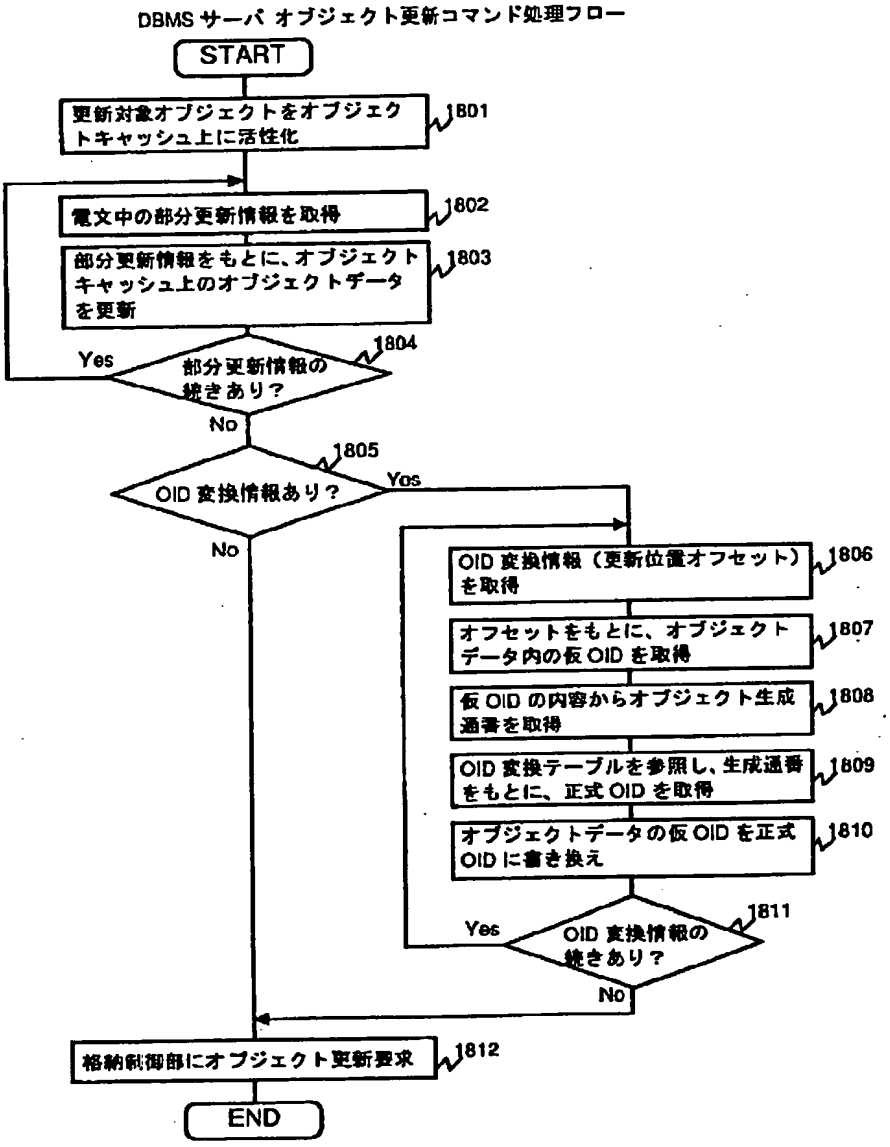
【図１５】


DBMS クライアント オブジェクト更新要求電文作成の処理フロー
START
部分更新情報チェインをたどり、更新要求数をカウントし、電文に必要なサイズを算出
1501
OID 変換情報チェインをたどり、変換する仮 OID 数をカウントし、電文に必要なサイズを算出
1502
更新要求電文全体に必要なサイズを算出し、通信管理部で電文領域を確保
1503
更新対象オブジェクトの OID を電文に設定
1504
部分更新情報チェインをたどり、部分更新情報取得
1505
部分更新位置オフセット、更新範囲、および更新データ内容を、電文のデータ領域にコピー
1506
部分更新情報の続きあり？
1507
Yes
No
部分更新情報の数およびオフセットをブロック制御ヘッダに設定
1508
当該更新データ中にOID 変換情報あり？
1509
Yes
No
OID 変換情報チェインをたどり、OID 変換情報取得
1510
OID 変換情報を電文に設定
1511
OID 変換情報の続きあり？
1512
Yes
No
OID 変換情報の数およびオフセットをブロック制御ヘッダに設定
1513
END

【図１６】


DBMSサーバ トランザクションコミットの処理フロー
START
通信管理部から電文のアドレスを取得
1601
電文中の変換OID総数をもとに、
OID変換テーブルを初期化
1602
電文からコマンドパラメタブロック取得
1603
1604
コマンド識別子が
「生成」？
Yes
No
生成コマンド処理
1605
1606
コマンド識別子が
「更新」？
Yes
No
更新コマンド処理
1607
1608
コマンド識別子が
「コミット」？
Yes
No
1609
トランザクション管理部へ
トランザクションコミット要求
END
次のコマンドパラメタブロックを取得
1610

【図１８】


DBMS サーバ オブジェクト更新コマンド処理フロー
START
更新対象オブジェクトをオブジェクトキャッシュ上に活性化
1801
電文中の部分更新情報を取得
1802
部分更新情報をもとに、オブジェクトキャッシュ上のオブジェクトデータを更新
1803
部分更新情報の続きあり？
1804
Yes
No
OID 変換情報あり？
1805
Yes
No
OID 変換情報（更新位置オフセット）を取得
1806
オフセットをもとに、オブジェクトデータ内の仮 OID を取得
1807
仮 OID の内容からオブジェクト生成通番を取得
1808
OID 変換テーブルを参照し、生成通番をもとに、正式 OID を取得
1809
オブジェクトデータの仮 OID を正式 OID に書き換え
1810
OID 変換情報の続きあり？
1811
Yes
No
格納制御部にオブジェクト更新要求
1812
END